4-2018

Design Considerations for Training Memristor Crossbars Used in Spiking Neural Networks

Lydia M. Hays
lmh7513@rit.edu

Design Considerations for Training Memristor Crossbars Used in Spiking Neural Networks

Lydia M. Hays

Design Considerations for Training Memristor Crossbars Used in Spiking Neural Networks

Lydia M. Hays

April 2018

A Thesis Submitted
in Partial Fulfillment
of the Requirements for the Degree of
Master of Science
in
Computer Engineering

R·I·T | Kate Gleason
College of ENGINEERING

Department of Computer Engineering

Design Considerations for Training Memristor Crossbars Used in Spiking Neural Networks

Lydia M. Hays

Committee Approval:

Dr. Dhireesha Kudithipudi *Advisor* Date
Professor

Dr. Eric Bohannon Date
Postdoctoral Researcher

Dr. Marcin Łukowiak Date
Associate Professor

Acknowledgments

First, my greatest thanks to my Heavenly Father for being a constant source of wisdom and serenity during my life.

I would like to thank my committee members: Dr. Kudithipudi for her support and guidance throughout this thesis and the next stage of my academic career, Dr. Lukowiak for his patience and understanding throughout all my delays, and Dr. Bohannon for his weekly meetings and invaluable analog advice as I learned this strange art-form.

I am also thankful for the Nanocomputing lab. Nick Soures and Abdullah Zyarah, thank you for helping me through my first papers and with various analog assistance. In particular, Nick, thank you for a being a source of calm and reassurance during a very anxious point in my life. For the whole lab, I valued your camaraderie and loved discovering exciting new foods during our pot lucks.

To my RIT friends: thank you for being in the trenches with me throughout classwork and thesis work. Late nighters were much more tolerable with friendship, pizza, and wings. Chris Culpepper, your honesty and crazy antics were always a breath of fresh air. Will Gowell, I will always treasure our semesters TAing together. And a final thanks to all my students who quickly became great friends.

Finally, I can never thank my family enough. Your support through this crazy five years has been invaluable. Thank you for your advice on life and homework, the great food and warm hugs, and reminding me that life is more than grades and check-marks on my resume. I love you and am forever blessed to be a part of this growing family.

To my wonderful family: my sanity, support, and entertainment.

I love you all.

Abstract

CMOS/Memristor integrated architectures have shown to be powerful for realizing energy-efficient learning machines. These architectures are recently demonstrated in reservoir computing networks, which have reduced training complexity and resource utilization. In reservoir computing, the training time is curtailed due to random weight initialization in the hidden layer, which will remain constant during training. The CMOS/memristor variability can be exploited to generate these random weights and reduce the area overhead. Recent studies have shown that the CMOS/memristor crossbars are ideal for on-device learning machines, including reservoir computing networks. An exemplary CMOS/memristor crossbar based on-device accelerator, Ziksa, was demonstrated on several of these learning networks.

While the crossbars are generally area and energy efficient, the peripheral circuitry to control the read/write logic to the crossbars is extremely power hungry. This work focuses on improving the Ziksa accelerator peripheral circuitry for a spiking reservoir network. The optimized training circuitry for Ziksa includes transmission gates, a control unit, and a current amplifier and is demonstrated within a layer of spiking neurons for training and neuron behavior. All the analog circuits are validated using the Cadence 45 nm GPDK on a 2x4 and 1x4 crossbar. For a 32x32 crossbar, the area and power of the peripheral circuitry is ~2,800 μm^2 and ~3.685 mW respectively, demonstrating the overall efficacy of the proposed circuits.

Contents

List of Figures

List of Tables

Acronyms

CTAT/PTAT

Complementary/Proportional to Absolute Temperature

ESN

Echo State Network

ICMR

Input Common Mode Range

LFSR

Linear Feedback Shift Register

LSM

Liquid State Machine

RC

Reservoir Computing

RNN

Recurrent Neural Network

SNN

Spiking Neural Network

Chapter 1

Introduction

1.1 Motivation

The human brain vastly outperforms modern computing in speed, computational complexities, and power usage. For years, software has advanced in mimicry of neural pathways to capture the advanced learning capabilities of the human brain. It started with the perceptron in 1958: a highly abstracted version of a biological neuron [6]. Connecting multiple perceptrons together can create artificial neural networks that, with advancements in training algorithms and architectures, have cemented themselves as the future of learning and data processing in modern computing. Recurrent neural networks (RNNs) take a step closer to the biological neural networks by adding recurrent connections that add a temporal aspect to the processing [7]. While powerful, the training time for these networks increases exponentially with an increase in size. An architectural solution is to use reservoir computing [8]. In reservoir computing (RC), the massive interconnect of neurons is initialized with random weights which do not change during training. Instead, only the output layer has its weights updated which cuts down on training significantly while allowing for large, interconnected networks.

While training only the output layer has reduced the training time, larger and larger networks are needed leading to research into digital and analog implementations

of the neural networks. RNNs are a highly generalizable solution to a multitude of machine learning problems because training for a new problem only requires retraining of the output layer, not a redesign of the architecture [8]. This makes it feasible to build the network in hardware to provide significant gains in speed. While the neurons can be made with simple circuits, the weights of the network need to be non-volatile. Every synaptic connection requires a weight and increasing the precision of the weights greatly increases the size of the network in traditional CMOS technology. Unlike with traditional Von Neuman architectures, neural networks need their data (the weights) to be intermingled with the computational hardware (the neural connections). One way to address this challenge is to use non-volatile memory devices like memristors that retain the weights and can store higher-precision bits due to multi-level storage [9].

The memristor was theorized by Leon Chua in 1971 as the 4th fundamental circuit element [10]. A memristor has a non-linear relationship between electrical charge and magnetic flux. There is debate on whether the experimental memristors available today can meet the requirements, however, they offer a unique property as a two terminal, non-volatile memory device [11]. The device remembers the previous current passed through it by changing its resistance in a hysteresis relationship between current and resistance. The simplest memristors offer single bit precision. Experimental memristor devices with six states are demonstrated in literature and theoretically they can offer infinite states [12]. This device, especially in consideration of the multiple states, offers a significant decrease in size required for memory storage. In the network, the memristor forms the connections inbetween neurons.

While the memristors themselves can be made very small, what determines the circuit usefulness is the size of the analog circuitry that surrounds the memristors. Depending on the architecture, the circuitry can be broken into four functions: the neurons, training the memristor weights, reading the results, and calculating the

new weights. Calculations can be accomplished in digital hardware which will use minimal space compared to the analog. Reading the results can be kept minimal by using spiking neurons that pass data as 1's and 0's so that a simple counter can read them. The neurons themselves have been extensively researched, but the circuitry that trains the memristors can easily be massive due to the multiple operational amplifiers required. The objective of the thesis is to provide and validate training circuitry for memristor crossbars in the Cadence 45nm GPDK.

1.2 Objectives

For this thesis, the focus is on the circuitry that trains the memristors. In particular, it is improving on the Ziksa training circuit in functionality and area usage [2]. This leads to the following objectives

1. Improve upon functional design issues in the Ziksa training circuit and the space required for its implementation.
2. Implement the new design in Cadence with the GPDK 45nm PDK. For each of the main analog components:
   (a) Justify the topology choice and the specifications needed for their design.
   (b) Pass these specifications over process, temperature, and supply voltage corners.
   (c) Pass these specifications over 1000 Monte Carlo runs at nominal voltage and temperature.
3. Show the new circuitry training memristors in positive and negative directions with minimal power usage.
4. Demonstrate the integration of the training unit within a spiking neuron for future use in spiking neural networks.

5. Analyze the area usage of the training circuitry within neural networks.

1.3 Document Structure

Chapter 2 overviews the background and related works with a brief overview of neural networks, memristor devices, existing research, and a break down of the Ziksa training circuitry and the problems with the design. Chapter 3 starts with the improvements to Ziksa and the high-level architecture. It then delves into the individual analog components, their topologies, and how the high-level architecture forces certain specifications. Chapter 4 is the results from high-level nominal simulations and in depth simulations of the individual analog components. Chapter 5 discusses the final thoughts for this research and how it can be used in future works for hardware neural networks.

Chapter 2

Background and Related Works

For years, computer performance and power has been improved by parallelism, pipelining, and moving to smaller CMOS technology nodes. Unfortunately, smaller transistor sizings are causing increasing difficulties as non-ideal behaviors overpower basic functionality [13]. This has encouraged a resurgence in neuromorphic computing using memristive devices that can provide an energy efficient alternative to traditional CMOS computing.

2.1 Neurons

Neuromorphic computing, often referred to as brain inspired computing, researches implementing neural networks in hardware. The high-level architectures are the same as used in Machine Learning, including two different common neurons, the first shown in Figure 2.1.

The first model of the neuron was the perceptron back in 1958 [6]. This is a highly simplified version of a biological neuron where the output is a linear combination of the input values (X) multiplied by weights (W). This is similar to an FIR filter with the taps being replaced by the input values of X. In a traditional perceptron, a simple threshold is used to determine if the output would be a 1 or a 0. More complicated activation functions such as the sigmoid shown in Figure 2.1 can be used. The output

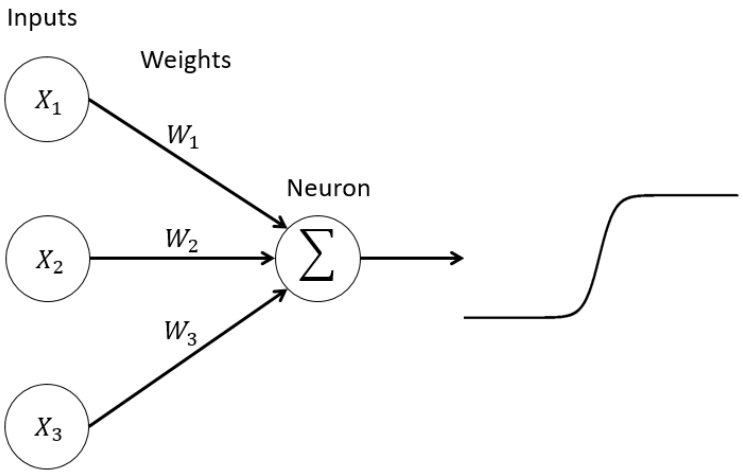

Figure 2.1: A single perceptron neuron

would then be as follows.

$$Y = sigmoid(\sum_{i=1}^{n}[X_i W_i]) \tag{2.1}$$

In hardware, the input values would need to be analog values which creates issues with accuracy and reading the values. Training requires reading the exact input value which would require at least one analog to digital converter (ADC). The ADC could occupy a significant amount of area. An alternative is a neuron that is closer to biological neurons shown in Figure 2.2.

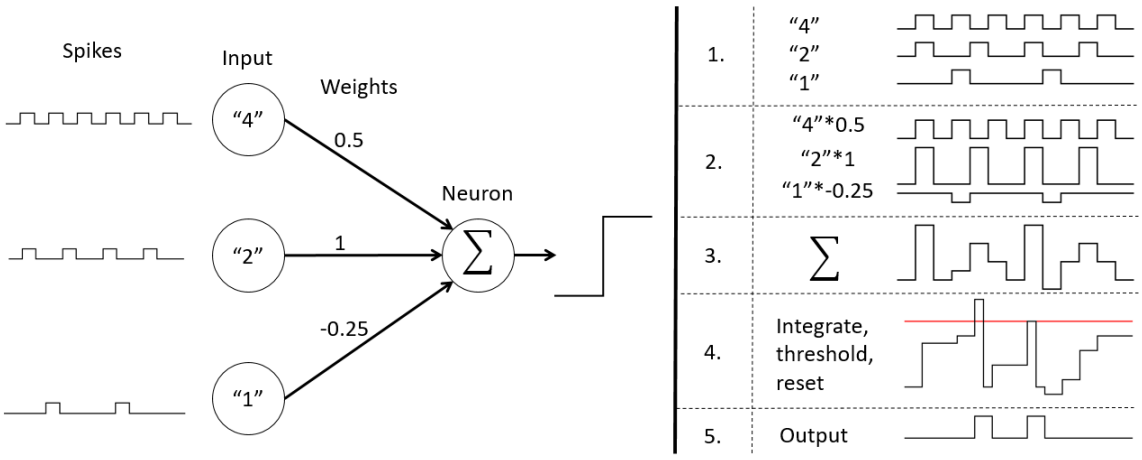

Figure 2.2: A leaky integrate and fire neuron

In the leaky integrate and fire (LIF) neuron, a type of spiking neuron, data is encoded into spikes of 1's and 0's. This solves this issue of ADCs because the data is inherently digital so a simple counter can be used. There are many ways of encoding data into spikes, such as rate or latency, but the LIF still functions the same [14]. The example on the right side of Figure 2.2 shows inputs encoded into spikes according to rate such that "2" is twice the frequency of "1" and "4" is twice the frequency of "2". In step 2, the inputs are multiplied against the weights of the neurons which changes the height of the spikes and some can become negative. Step 3 shows the result of summing the three multiplied inputs while step 4 shows the internal 'voltage' of the neuron. In an LIF neuron, the values from step 3 are integrated over time into this voltage. If the voltage passes a set threshold, represented as a red line, the voltage resets to 0 V, fires its own spike seen in step 5, and begins to integrate again.

Two things not shown in this graphic are the leaky part of an LIF neuron and the refractory period. Both would occur in step 4. In an LIF neuron, there is a constant leakage of the internal voltage over time. This allows the neuron to 'forget' old data if it hasn't received spikes in a long time and prevents very low inputs from firing. The refractory period occurs after a spike has been triggered. The neuron must take a short period of recovery time and will not integrate or fire another spike until the recovery period is over. Stringing perceptrons together creates traditional artificial neural networks (ANNs) while combining LIF neurons creates a spiking neural network (SNN). For this thesis, all networks created are SNNs.

2.2 Neural Networks and Reservoir Computing

Connecting perceptron or LIF neurons together can create the neural networks shown in Figure 2.3.

A feed-foward networks network only connects the neurons in one direction and has no memory of previous inputs. A recursive network adds in backwards connec-

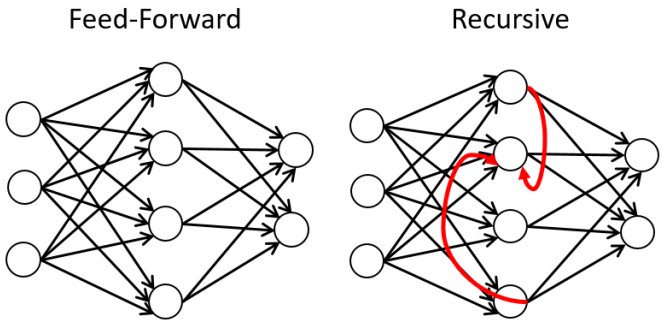

Figure 2.3: Feed-foward and recursive neural networks

tions that give it a memory of previous inputs. This allows for the processing of time series data, but increases training time significantly. To decrease training time architecturally, Reservior Computing (RC) networks can be used.

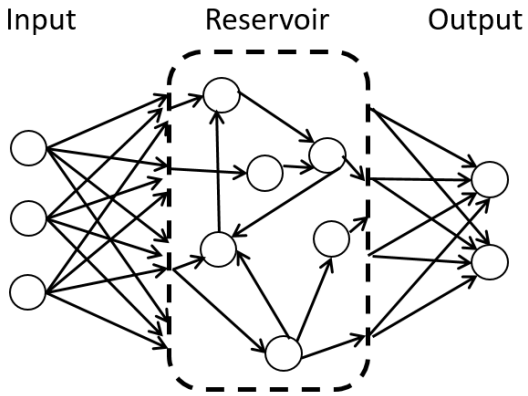

Figure 2.4: A high level view of the reservoir computing architecture

The architecture can be broken down into three parts: the input layer, the recurrently connected reservoir, and the output layer. The recurrent connections in the reservoir are what give RC architectures the ability to classify temporal data and add massive complexity to training. To save training time, the reservoir weights and the weights from the input layer are never trained. By only training the output layer, the back propagation can be simplified to a single linear equation. The downside is that to achieve a similar accuracy as a fully trained network, the reservoir must be made large.This leads to a large amount of weights which becomes a major issue in

neuromorphic computing.

Reservoir Computing (RC) is used for solving temporal classification, regression, or prediction tasks such as weather prediction, system control, or speech recognition [8] [15]. There are two main types of RC: Echo State Networks (ESN) and Liquid State Machines (LSM). ESNs often use a standard sigmoidal neuron and depend on a linear combination of the output of the reservoir to determine the output [16]. LSMs are closer to the biological inspiration by using LIF neurons [8]. The neurons and training circuits for this thesis are designed to be able to work on feed-forward or RC networks.

2.3 Memristor

The memristor, also known as a memory resistor, was theorized in 1971 by Leon Chua [10]. It is a passive, two terminal element with a state-dependent ohm's law. A memristor is a thin semiconductor film sandwiched between two metals as shown below [11].

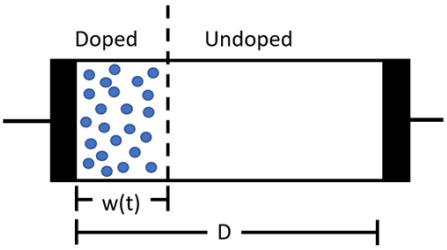

Figure 2.5: The different doping regions of a memristor

One end of the memristor has a much higher doping concentration than the other where the resistance is determined by

$$R(t) = R_{doped}\frac{w(t)}{D} + R_{undoped}(1 - \frac{w(t)}{D}). \tag{2.2}$$

A higher doping concentration leads to a lower resistance. If a voltage is applied

across the memristor that exceeds a device threshold, the ratio of doped to undoped regions will change according to the magnitude, polarity, and length of time the voltage has been applied such that

$$\frac{\delta w(t)}{\delta t} = \mu_v \frac{R_{doped}}{D} i(t) \tag{2.3}$$

where μ_v is the doping mobility. In general, increasing the resistance and decreasing the resistance will occur at different rates. This is because when training to a lower resistance, the diffusion forces and applied electric field act in the same direction. When training to a higher resistance, the applied flied is opposite of the Fickian diffusive force causing the doping boundary to move at a slower rate [17]. The memristor follows a hysteresis curve that varies greatly depending on the material and process.

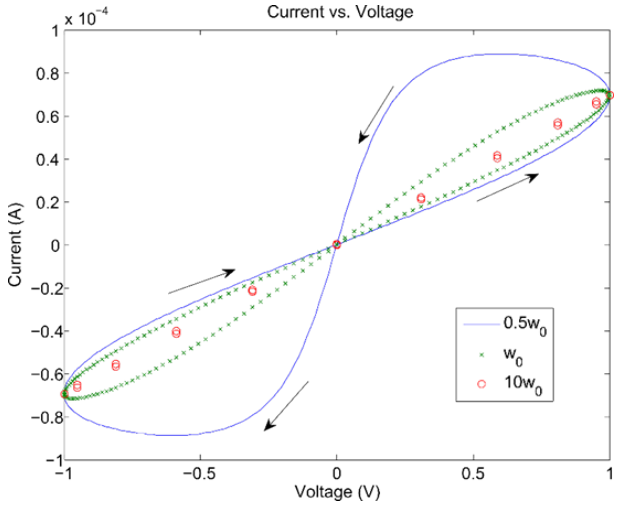

Figure 2.6: The pinched hysteresis loop of a memristor [1]

Figure 2.6 shows that once the voltage passes a high threshold, the resistance state changes so that as the voltage drops, there is a different slope relating current and voltage. w_o is the frequency at which the signal is being passed through the memristor. When it reaches higher frequencies, the memristor's resistance changes at a lower rate until it behaves like a linear resistor.

The first memristor was demonstrated by HP labs in 2008 [11]. Memristors can be made with $SrTiO_3$, TiO_2, NiO, and others with varying resistance ranges from hundreds of Ohms to Gigaohms [18]. Whatever resistance the memristor is set to will hold when not powered allowing memristors to be used as non-volatile memory. A typical use is to force the memristor to its high resistance state (HRS) or low resistance state (LRS) leading to a 1-bit memory. Memristors can also be set to the in between resistances, theoretically providing infinite states.

2.4 Memristor Crossbar

To use the memristor in RC, a common architecture for the neurons and their connections is the memristor crossbar shown in Figure 2.7 [9].

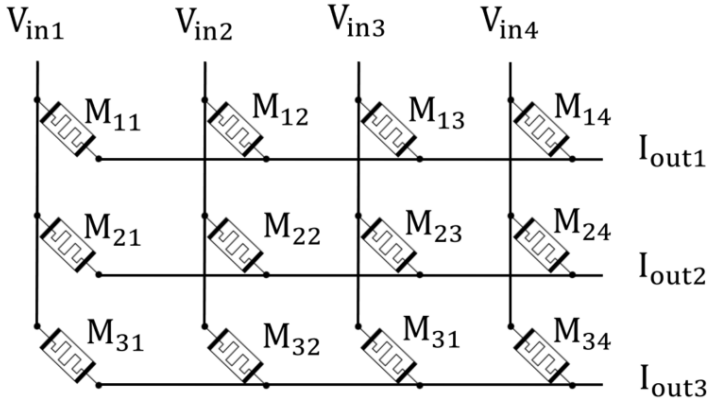

Figure 2.7: A 3x4 memristor crossbar

The inputs enter through the top of the crossbar as either analog values for perceptrons or as spikes, ranging from 0 to 1 (with 0 and 1 being any voltage required) for spiking neurons. If the output nodes are held at 0 V, the output currents can be represented as

$$I_{out1} = \frac{V_{in1}}{M_{11}} + \frac{V_{in2}}{M_{12}} + \frac{V_{in3}}{M_{13}} + \frac{V_{in4}}{M_{14}} \tag{2.4}$$

where M_{XY} is the resistance of the memristor. The memristor resistances behave like the weights in a neural network with a high resistance equating to a low weight. The crossbar allows for a compact design when using a highly interconnected network. These outputs can then be fed into different activation functions for perceptrons or LIF neurons. One crossbar represents a single layer in a neural network and completes the expensive matrix multiplication through analog computation. Multiple crossbars can then be combined to create feed-forward networks of RC architectures.

2.5 Related works

Using memristor crossbars is a well researched area with proposed architectures published before HP labs produced a working memristor [19]. The first designs focused on using crossbars purely for condensed non-volatile memory storage due to its high density of 154 to 309 Gb/cm^2 making it 2-4x as dense as a hard drive when used as single bit memory [20] [21]. Unfortunately, crossbars have a limit in size due to current sneak paths affecting the accuracy of reading and training [22]. An equivalent number of memristors, but at lower density, can be accomplished by chaining multiple crossbars together. This is the approach used for current neuromorphic systems [23].

For these neural networks, the circuitry around the crossbar is vital to preform training, reading, and isolation of the different crossbars. Some require transistors inside the crossbar for every memristor that work as switches for training and reading [24]. Others require multiple op-amps for every row and column such as [25] which require amplifiers and DACs for training. The Ziksa training unit utilizes only one amplifier per neuron, but has other set backs that will be improved upon in this thesis [2]. The ziksa circuitry can be seen in Figure 2.8.

The input voltage and *VDD/2* are set to be less than the threshold voltage of the memristor to avoid training. During a read, the *ZRow* and *ZCol* circuits are turned off and a read signal allow the input signals through the columns. When a

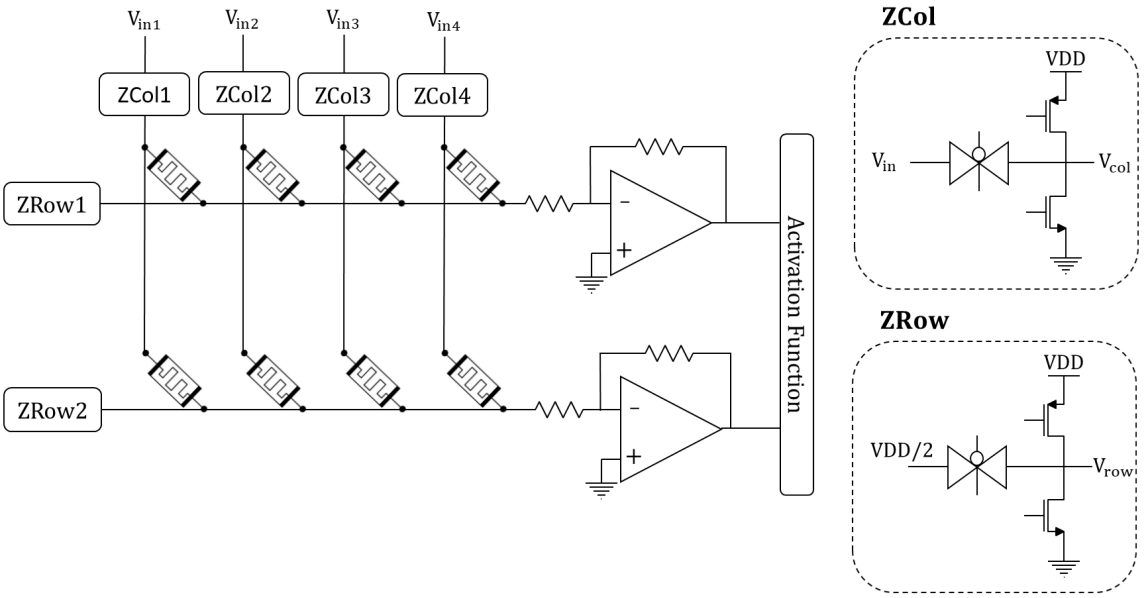

Figure 2.8: The Ziksa training unit [2]

memristor is being trained, the *ZRow* and *ZCol* surrounding that memristor are set to *VDD* and *GND* so that the memristor will train. All other columns are set to *VDD/2* and all other rows have their *ZRow* circuits disabled so that the row floats. This prevents the other memristors from having a voltage across them greater than the threshold voltage so they will not train. If the memristor is being trained in the positive direction, the *ZCol* will be set to *VDD* and the *ZRow* set to *GND* to provide a voltage greater than the threshold voltage. For training in the negative direction, the training units switch their voltage. This design presents two main problems: memristor weight linearity and current draw during training.

The first problem was introduced due to the *ZRow* circuit. Since it forces the row to *VDD* or *VDD/2*, a resistor was needed between the crossbar and the negative input of the differential amplifier since the negative input is forced to *GND*. Because of this

resistor, the output current during a read of four memristors would be expressed as

$$I_{out} = \frac{V_{in1}M_2M_3M_4 + V_{in2}M_1M_3M_4 + V_{in3}M_1M_2M_4 + V_{in4}M_1M_2M_3}{1 + R(M_2M_3M_4 + M_1M_3M_4 + M_1M_2M_4 + M_1M_2M_3)} \tag{2.5}$$

where V_{inX} is the input signal, M_X the resistance of the memristor, and R is the the resistor between the amplifier and the crossbar. The main issue with this equation is that the memristor weights no longer follow the neuron behavior of linear independence seen in Equation 2.4. Instead, to increase the weight of one memristor, all other memristors must be taken into account.

This resistor also causes the second issue of current draw during training. When training a memristor in either direction, all other columns are forced to *VDD/2* causing a constant *VDD/2* over the R. When training a memristor in the negative direction, that column is forced to *VDD* causing twice the current draw. A large resistor will reduce the current, but needs to be sized in relation to the memristor resistance range because of Equation 2.5.

2.6 Robustness of Memristor Neural Networks

A key concern with analog design is how much leeway is allowed with certain specifications. This is discussed in detail in [26] through the Matlab simulation of a Liquid State Machine (LSM) with memristor crossbars. The focus was on the variability of memristor read and write times. The general conclusion was that if the noise was inherent to the device and did not change over the course of training, the network was robust to read and write noise due to the training algorithm adjusting the memristor weights. If the noise varied while it was training, the accuracy was seriously affected. With this in mind, circuits such as the voltage and current reference that drive training and reading can not significantly vary over time, temperature, or voltage, but

variation is allowed across process variation as this would be fixed with the training of the neural network.

Chapter 3

Circuit Designs

3.1 High-Level Design

While the focus of this thesis is on the training circuitry, the functionality must also be demonstrated in the context of a fully functioning neuron. The circuits built for this thesis create a layer of spiking neurons that have the following functionality:

1. Randomly initialize weights
2. Possess the ability to train individual weights
3. Use a spike train for the neuron inputs
4. Multiply the inputs against the weights and sum them
5. Use a leaky integrate and fire neuron to determine output
6. Produce spike trains of varying frequency based on input frequency and memristor weights
7. Can be connected to other columns of spiking neurons in a feed-forward or recursive network

The high-level circuit design can be seen in Figure 3.1.

The Ziksa training circuitry (ZTC) satisfies points 1 through 3 while the crossbar satisfies point 4. The inverting and non-inverting inputs of the current amplifier are

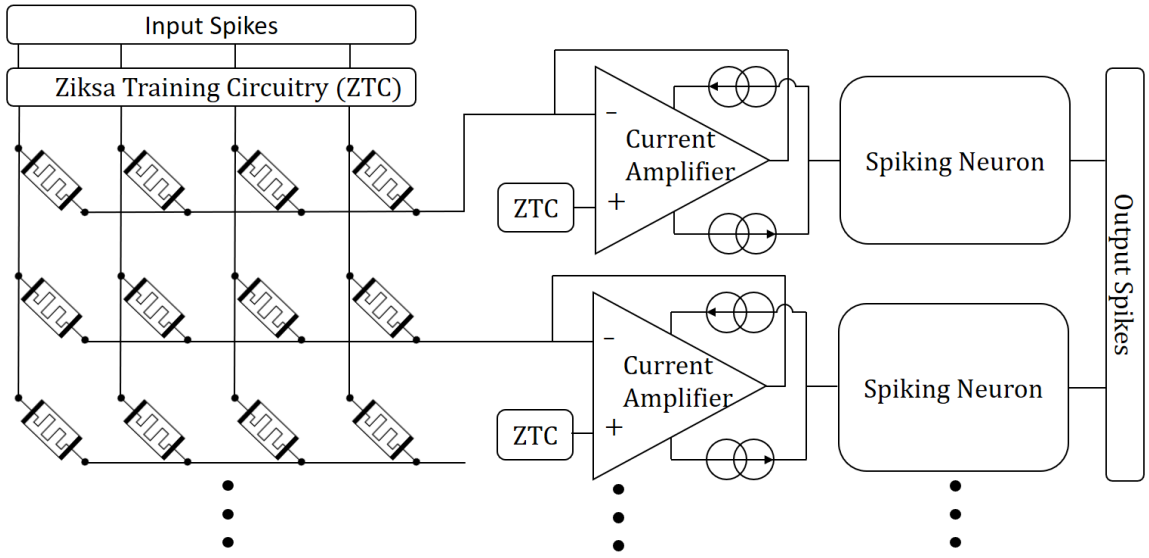

Figure 3.1: A high-level overview of the spiking neurons

necessary for ZTC to function and force the outputs of the crossbar to 0 V so that the memristor weights remain linearly independent. The current amplifier passes the current into the spiking neuron which satisfies points 5 through 6. All the output spikes are clocked to avoid delay issues and are buffered from the next layer of neurons with SR latches to satisfy point 7.

3.2 Arbitrary Constraints

There are a few constraints to the system that were arbitrarily chosen to begin the design process. The rails of the system were set to 1 V and -1 V with a GND rail available as well. This provides more headroom to work with positive and negative crossbars in future designs. The 2 V rails also encourage the use of the GPDK45 2 V devices in topologies where the transistors are minimally stacked. The downside is that these devices have thicker oxide layer leading to a decreased transconductance according to

$$g_m = \mu_n C_{ox}(\frac{W}{L})(V_{gs} - V_{th}) \tag{3.1}$$

because C_{ox} is inversely proportional and V_{th} is proportional to the oxide thickness. The latter is due to

$$V_{th} = V_{TO} + \gamma(\sqrt{|V_{SB} + 2\phi_f|} - \sqrt{|2\phi_f|}) \tag{3.2}$$

where V_{TO} is the threshold voltage with zero substrate bias, V_{SB} is the source to body substrate bias, $2\phi_f$ is the surface potential and γ is the is the body effect parameter

$$\gamma = (\frac{t_{ox}}{\epsilon_{ox}})\sqrt{2q\epsilon_{si}N_A} \tag{3.3}$$

where t_ox is the oxide thickness, ϵ_{ox} is the permitivity of oxide, ϵ_{si} is the permitivity of silicon, q is the elementary charge, and N_A is the doping concentration.

The next constraint was to set the positive input signal to a square wave spike of 0 V to 1 V. This allowed spikes to easily work with digital logic at the 45nm scale which uses rails from 0 V to 1 V. If negative spikes are used, they would range from 0 V to -1 V and would be level shifted. The width of the spikes was set to 25 ns to match the RC constant of the spiking neuron that is discussed in Section 3.6.

The memristors were set to have resistances from 200 kΩ to 1 MΩ with a crossbar of 2x4. Therefore, the highest current through the system would be 4 memristors in parallel(50 kΩ) leading to 20 μA for a 1 V spike. Though training will have 1.2 V across, only 1 memristor can be trained at a time so the highest current needed would be 6 μA from a single amplifier.

Of these constraints, the rails and the spike characteristics are unlikely to change when used for other applications. The memristor values and the size of the crossbars could vary depending on the devices being used and the network needed to solve a classification problem. The only circuit that should need to change would be the current amplifier. While the amplifier could technically be designed to work over a

wide range of devices, that could lead to a large, over-designed op-amp.

3.3 Improving The Ziksa Crossbar

The first major improvement was to move the row control to the positive input of the differential amplifier as seen in Figure 3.2 [3].

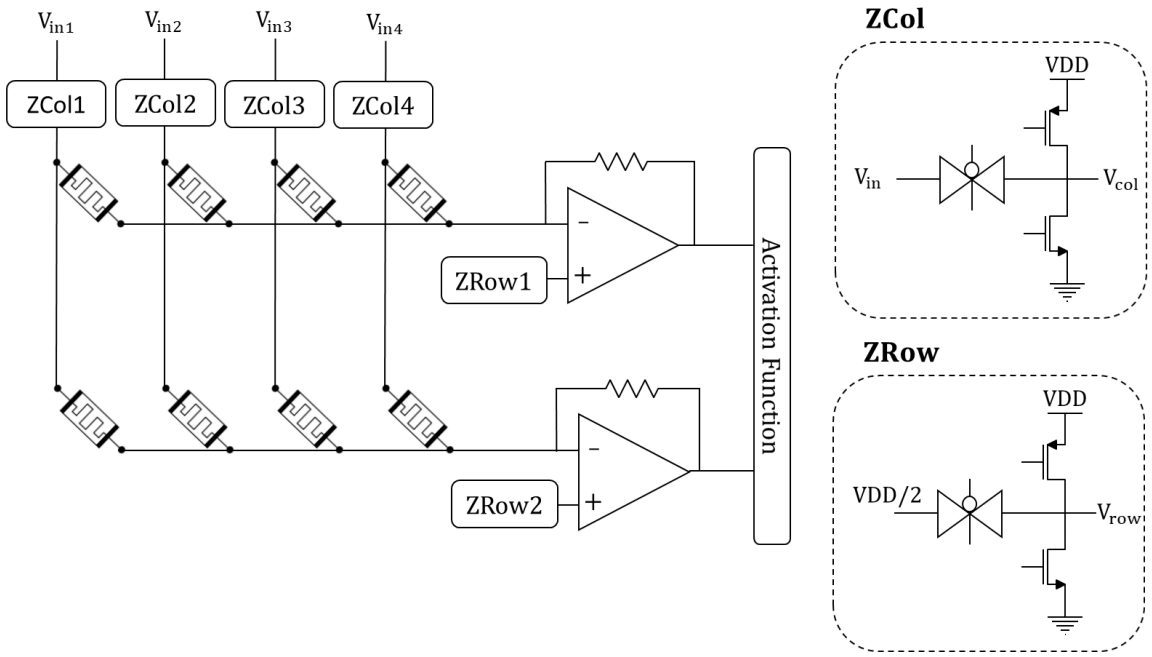

Figure 3.2: Memristor crossbar with the first improvement to the Ziksa training circuitry [3]

Since the output of the differential amplifier would be a voltage, the plan was to follow it with a voltage to current converter (another amplifier) and then dump the current into the spiking neuron as shown in Figure 3.1. In simulation with an ideal differential amplifier, this functioned perfectly so a simple two stage differential amplifier was designed and substituted in. Unfortunately, the amplifier was unable to force the negative input of the amplifier to VDD or GND so training could not occur. The original thought was that the issue was the amplifier being unable to sink and source enough current so the amplifier was redesigned to use an AB output stage. After a few different topologies were attempted, this too was unable to force

the column to the necessary voltage.

The problem was finally narrowed down to the Input Common Mode Range. ICMR dictates the range of input voltages under which the amplifier can meet the desired specification. For this design, the specification was to keep all transistors within saturation. While topologies exist that allow rail-to-rail ICMR , they are often very large[27]. Instead, the ideal amplifier was replaced and the memristors tested to determine the lowest V_{thmem} that would allow training and prevent the memristor from changing during a normal read. With the input signals from 0 V to 1 V, the lowest was a V_{thmem} of 1.05 V with the training at 1.1 V. Since this was with ideal amplifiers, it was decided to push to a V_{thmem} of 1.1 V and train at 1.2 V to give more room for variation. This required a slight change where certain transistor control switches were replaced with transmission gates because they would no longer be passing rail values. At the same time, it was determined that the differential amplifier and the voltage to current converter could be combined into a single-ended current amplifier. The end result is the Ziksa training circuitry seen in Figure 3.3 on a 2x4 crossbar.

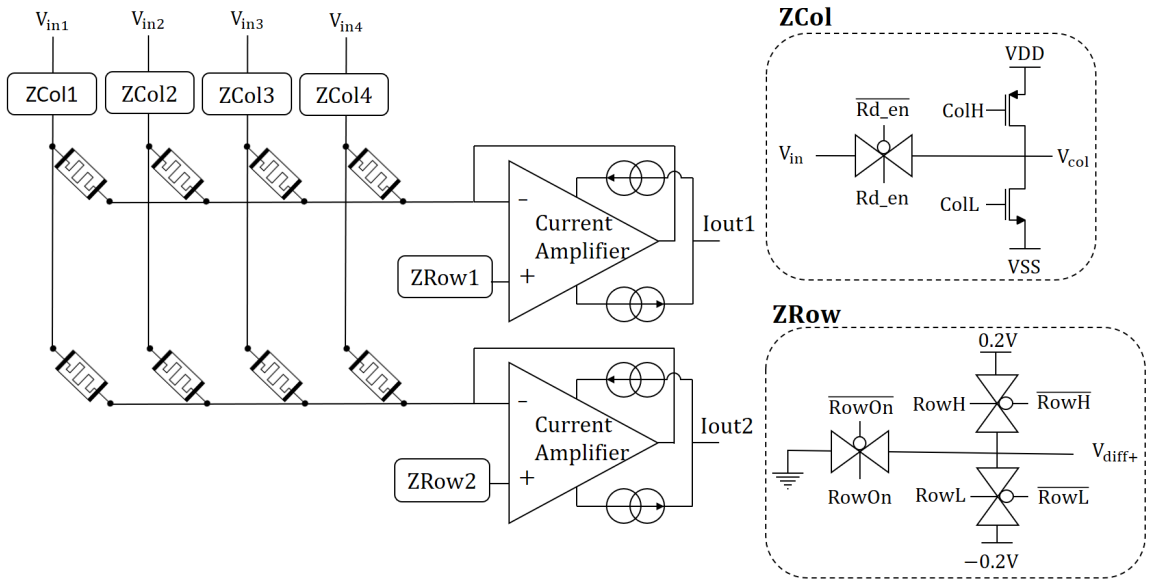

Figure 3.3: Memristor crossbar with Ziksa training circuitry

Each column has it's own ZCol, and each row it's own ZRow, circuitry with individual control signals. For ZCol, *Rd_en* controls whether the input voltage is passed through while *ColH* and *ColL* set the column to 1 V or -1 V during training. For ZRow, *RowOn* sets the row to 0 V during reading and when a different row is training. *RowH* and *RowL* set the row to 0.2 V and -0.2 V when training its memristors. There are three modes of operation: read, train high, and train low. During the read operation, no memristive device can have a voltage across it greater than V_{thmem}, which is set to 1.1 V. Figure 3.4 shows the voltages across a single memristor during training and reading.

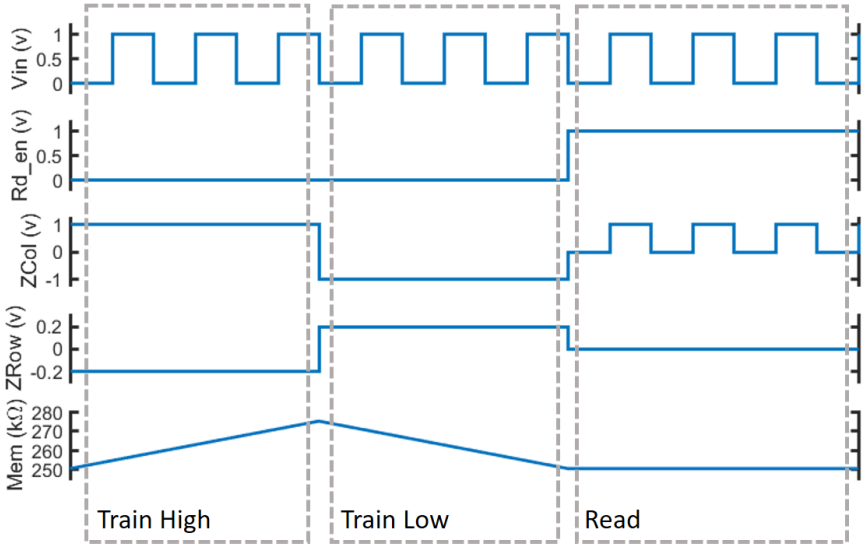

Figure 3.4: Voltages and memristor resistance during training and reading

During training, read is disabled to isolate the crossbar from the behavior of other neurons. When training high, the control signals set ZCol to 1 V and ZRow to -0.2V so that there is 1.2 V across the memristor. The voltage is greater than V_{thmem} thus increasing the memristor's resistance. The opposite occurs during training low so that the resistance will be decreased. During this training, the unused rows will have their ZRow set to 0 V so that it is impossible for any memristor on that row to have a voltage across greater than the threshold voltage. Unused columns will have their

ZCol completely turned off so that the node floats and will never supply the current to train a memristor, even if it is in the same row as a memristor being trained. To change the amount a memristor is trained, simply vary the length of time the voltage is applied. During read, the Zrow is set to 0 V and *Rd_en* is set high so that Vin can pass through to the resistor. Since the input spikes are set from 0 V to 1 V, they will never trigger a memristor to train.

This design fixes the issues with the old Ziksa crossbar. By removing the resistor between the amplifier and the crossbar, the memristor weights are once again linearly independent and power is saved because there is no longer a voltage forced across the resistor during training. Finally, size is saved due to the removal of the resistor between the crossbar and differential amplifier and a second resistor is removed because of the combination of the amplifier and voltage to current converter.

3.4 Control Unit

The Ziksa control unit initializes and trains the memristive devices. The general order is to train all memristors to their lowest value then randomly initializes them by training high for a period of time determined by an LFSR (Linear Feedback Shift Register). A simplified state machine can be seen in Figure 3.5.

The reset stages train all the memristors down to their lowest weight using a preset max value that is sufficient to train a memristor from the highest possible value to the lowest. Since all are being trained for the same time and each column has it's own op-amp, multiple rows can be trained at the same time. The reset stage simply sets *col* to the number of columns it will be training over. In the *resetStart* stage, the outputs are set to train the memristors low except the rows remain off until *resetMem*. This is because switching to full training causes a large current spike which can alter the weights of the memristors. Splitting the transition to training between two states removes the current spike and adds minimal time in comparison

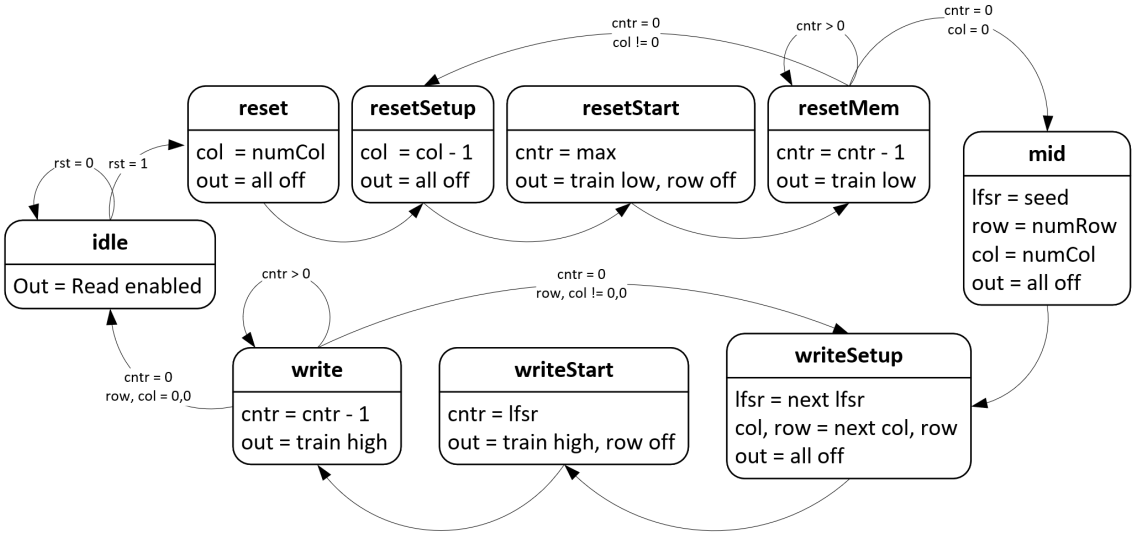

Figure 3.5: Control unit for initializing the memristor weights

to the length of training.

Once all memristors are set to their lowest value, *mid* sets up the LFSR for randomly initializing the memristors. The LFSR is sized so that its highest values will train a memristor to it's highest resistance. Due to the interconnect of the crossbar, the write stages must train only one memristor at a time so the stages *writeSetup*, *writeStart*, and *write* are repeated row by row, for every column in those rows. For each memristor, the next value generated by the LFSR is used to determine how long the memristor will train. Once all have been trained, it returns to *idle* ,which allows data to pass through the crossbar.

3.5 Current Amplifier

For the Ziksa training unit, a single differential input current amplifier is needed per row. The driving constraints of the op-amp, from Section 3.2, are that it must be able to sink/source 20μA of current, have an input common mode range (ICMR) of -0.2 V to 0.2 V, and a current gain of -1 A/A. The topology chosen can be seen in Figure 3.6

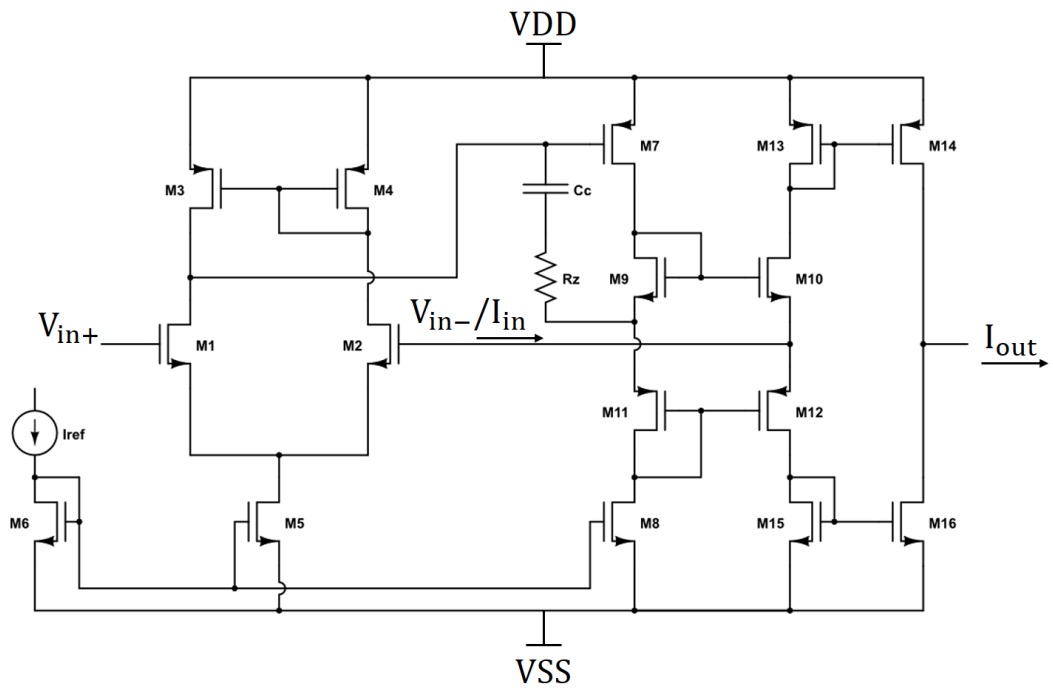

Figure 3.6: Differential amplifier with current output

While the op-amp is used for current gain, the voltage gain of the feedback loop determines the accuracy of the current gain. The gain can be expressed as

$$A_v = g_{m1}(r_{o1}||r_{o3})g_{m7}(r_{o7}||r_{o8})(\frac{g_{m10}/g_{m15}}{1+g_{m10}/g_{m15}}) \tag{3.4}$$

where $g_{m1}(r_{o1}||r_{o3})$ is the gain of the differential amplifier, $g_{m7}(r_{o7}||r_{o8})$ is the gain of M7, and $\frac{g_{m10}/g_{m15}}{1+g_{m10}/g_{m15}}$ is the gain of M10. The offset of the amplifier is determined by the matching of M1/M2 and the matching of M3/M4. Matching is improved with larger transistors and layout techniques. The ICMR is determined by how high and low the inputs can be pushed without pushing the transistors out of saturation. Since the voltage of V_{in+} is equal to the voltage of V_{in-}, the ICMR can be determined by either the differential portion or the current mirror stacks, depending on sizing. This can be seen in (3.5) and (3.6).

$$ICMR_{max} = min(VDD - V_{dsSat4} - V_{th2}, VDD - V_{dsSat13} - V_{th13} - V_{dsSat10}, VDD - V_{dsSat7} - V_{dssat10} - V_{th10}) \quad (3.5)$$

$$ICMR_{min} = max(VSS + V_{dsSat5} + V_{gs1}, VSS + V_{dsSat15} + V_{th15} + V_{dsSat12}, VSS + V_{dssat8} + V_{dssat12} + V_{th12}) \quad (3.6)$$

The required range is -0.2 v to +0.2 v which is possible because the rails run from -1v to 1v.

The current gain is determined by the ratio of the current mirrors at the output. If I_{in} is positive, it will be reflected across M15/M16 forcing I_{out} to be negative. A negative input current will reflect across M13/M14 causing a positive output current. The gain can then be expressed as

$$\frac{I_{out}}{I_{In}} = -(sgn(-I_{in})(\frac{(W/L)_{14}}{(W/L)_{13}}) + sgn(I_{in})(\frac{(W/L)_{16}}{(W/L)_{15}})) \quad (3.7)$$

where the function $sgn()$ is 1 for positive values and 0 for negative values.

A problem arises at the output due to the spiking neuron configuration. The output voltage will move as the capacitors charge up. Operating at the rails would cause issues due to M14 and M16 going out of saturation, so the output node is set to operate between 0 V and -0.3 V. With ideal transistors, as long as V_{gs} remains constant, the provided current should remain constant while in saturation. In reality, the curves follow Figure 3.7.

As V_{ds} increases, the output current will increase as well. A higher V_{gs}, until it reaches the limiting current, causes a larger change in drain current. This behavior

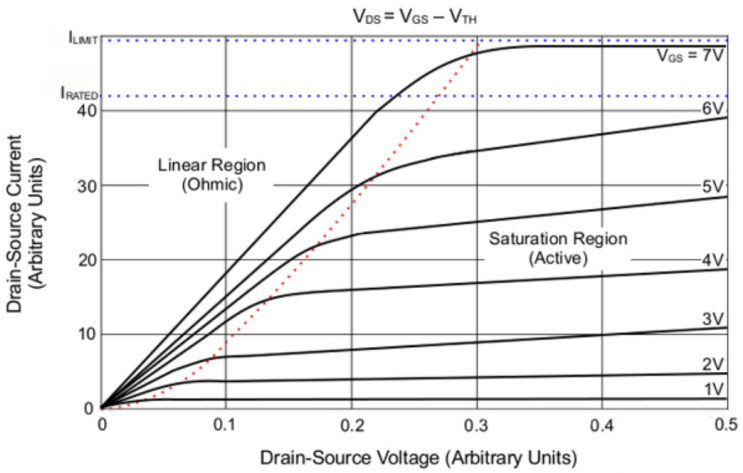

Figure 3.7: The arbitrary current supplied over increasing V_{ds} with varying V_{gs} [4]

gets worse with smaller technology nodes such as the GPDK45. For this topology, The solution is larger transistors and smaller V_{gs} leading to a smaller current. The downside is that a small V_{gs} can cause worse Monte Carlo results because there is less voltage headroom before the transistor will enter the linear region of operation. The smaller current would also increase the resistance of the transistor because resistance is inversely proportional to current. A high resistance becomes an issue with the poles of the amplifier. Another solution would be to change the topology to use a cascoded output layer. This would provide an increased output resistance so that as the voltage changes, the current change will be smaller.

Poles occur at every node where a signal passes through and can be approximated with

$$pole = \frac{1}{2\pi RC} \tag{3.8}$$

where R and C are the resistance and capacitance at the node being analyzed. To get a high phase margin, the poles must be manipulated so that there is a single dominant pole and the second pole does not occur until after the gain is less than one. Technically, any two pole system with a phase margin greater than 0° will be stable,

though it will oscillate for a long time. The oscillations will decrease in duration until approximately 60°, at which point any higher phase margin will be over-damped causing it to settle slowly, but without oscillation. Figure 3.8 shows what the bode plots look like as compensation is added.

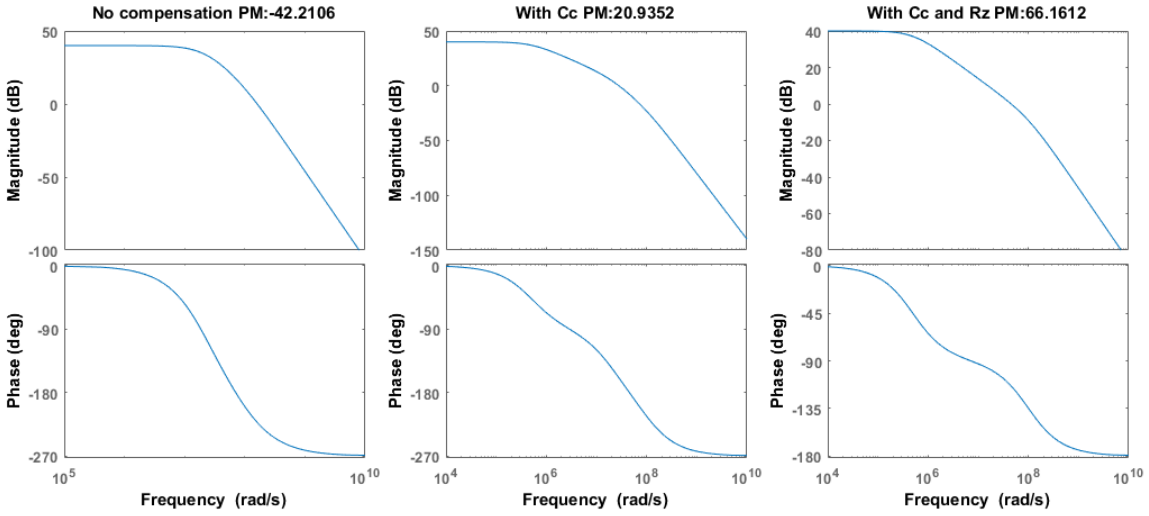

Figure 3.8: Bode plots with varying levels of compensation

The first set of bode plots shows a system with three poles that are very close together causing a low phase margin and an unstable system. These poles can be moved by adding adding C_C and R_Z in Figure 3.6. The compensation capacitor, C_c, utilizes Miller capacitance to increase its effective capacitance so that

$$C_{eq} = C_c(1 + A_v) \tag{3.9}$$

where A_v is the gain across the capacitor [28]. This forces one pole to become dominant and move to left as seen in the second graph in Figure 3.8. The phase margin has improved significantly but is still very low because the second and third poles are close together. Adding R_z creates a zero that can be adjusted to cancel out the secondary pole as seen in the third set of bode plots. Where equations can not inform an exact size, such as the ratios in Equation 3.7, the starting transistor sizes

were kept small to avoid low poles.

3.6 Spiking Neuron

The resulting current from the op-amp then enters a spiking neuron shown in Figure 3.9.

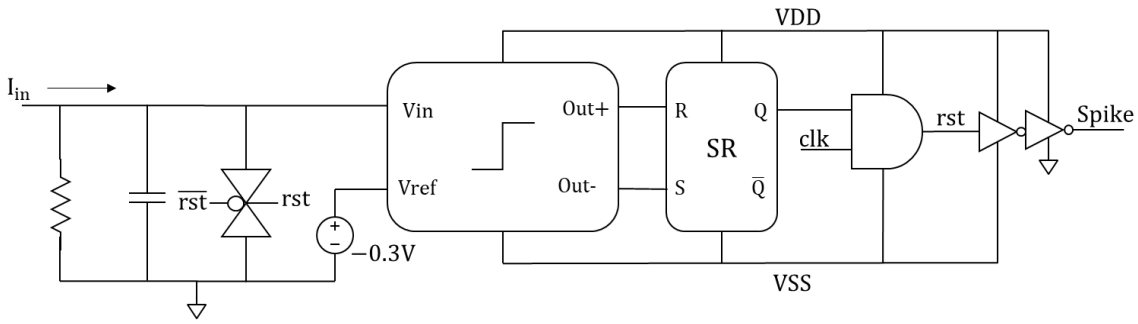

Figure 3.9: A spiking neuron that takes in current and outputs spikes of 0 V - 1 V [5]

The capacitor builds up charge and the resistor allows for leakage to mimic the biological leaky integrate and fire neurons. Since the op-amp has a current gain of -1 A/A, the capacitor charges from 0 V down to -0.3 V. Figure 3.10 shows the different nodes interacting while the neuron charges, fires, and discharges.

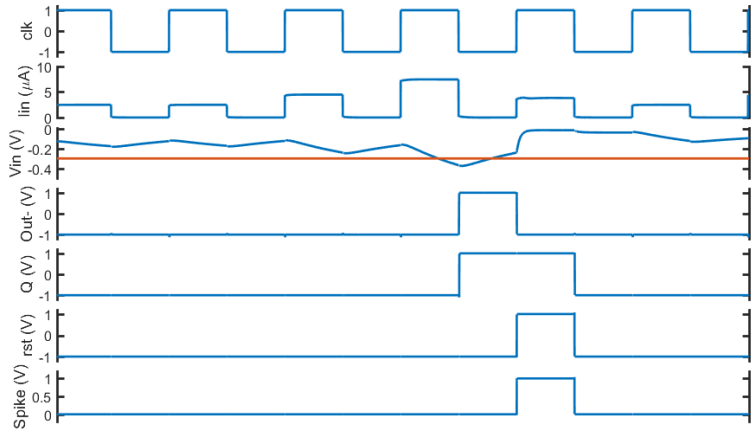

Figure 3.10: The different nodes of the spiking neuron in Figure 3.9

V_{ref} is set to -0.3 V and is represented as the red line in the V_{in} plot. Once V_{in}

reaches -0.3 V, the comparator's negative output will fire high. The SR latch holds onto the result and is ANDed with the 50 ns clk signal so that the spike width will only be 25 ns. The *rst* signal ranges from -1 V to 1 V and triggers the transmission gate so that the capacitor will be discharged to 0 V. The two inverters after the *rst* signal are used to level shift the spikes lower voltage from -1 V to 0 V so that it will be prepared for the next crossbar. The comparator is a double tail latched comparator from [29] shown in Figure 3.11.

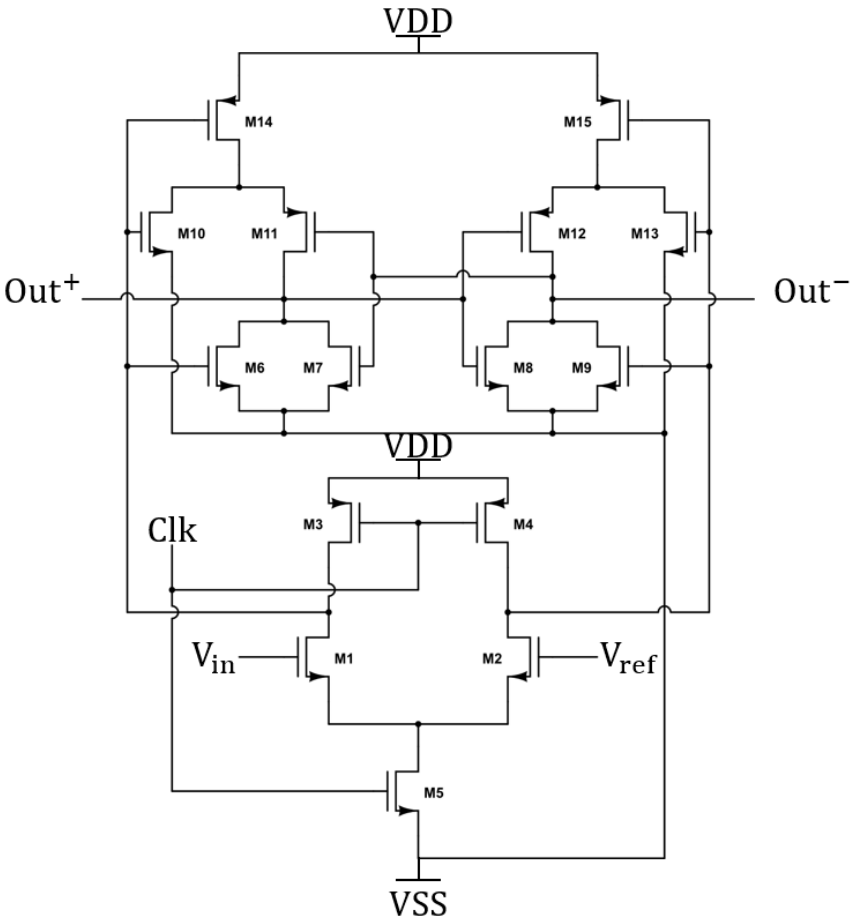

Figure 3.11: Double tail latched comparator

The basic operation is that when the clock switches high, the differential amplifier made of M1 - M5 will output a high enough difference that the latch, made of M7, M8, M11, and M12, will grab onto the difference and force the output to the rails. The important internal nodes can be seen in Figure 3.12.

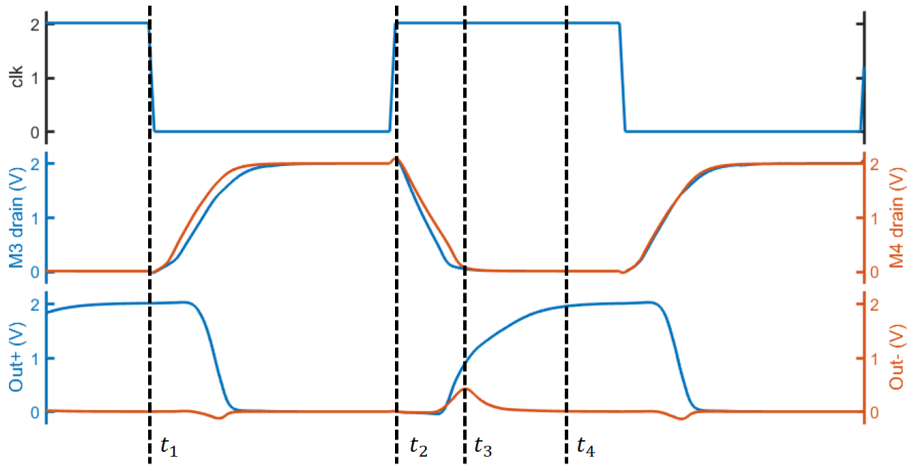

Figure 3.12: The timing of internal nodes in the comparator

At t_1, the clock is low so M3 and M4 pre-charge the top of the diff-amp to VDD while M5 is turned off so no computations can occur. Once the clock goes high at t_2, V_{in} is higher than V_{ref} so the drain of M3 drops slightly faster than the drain of M4. This pull of current starts to push the latch so that $Out+$ rises higher than $Out-$. Feedback kicks in at t_3 and it latches onto the difference and pulls to the rails by t_4. The key to functionality is that by t_3 there is a large enough difference between the two outputs that the latch can decide correctly.

For this to occur fast enough, the gain of the diff-amp must be high so that the difference between the sides will be large enough for the latch to detect. Also, the time constants of the the diff-amp and the latch must be small so that the nodes can charge quickly. Finally, the output loads of $Out+$ and $Out-$ must be equivalent for the nodes to have the same time constant and charge equivalently. Having the comparator followed by a SR latch assures that the output nodes will always have the same load.

3.7 Voltage and Current Reference

The reference voltage for the comparator is supplied by a Banba bandgap voltage reference shown in Figure 3.13 [30].

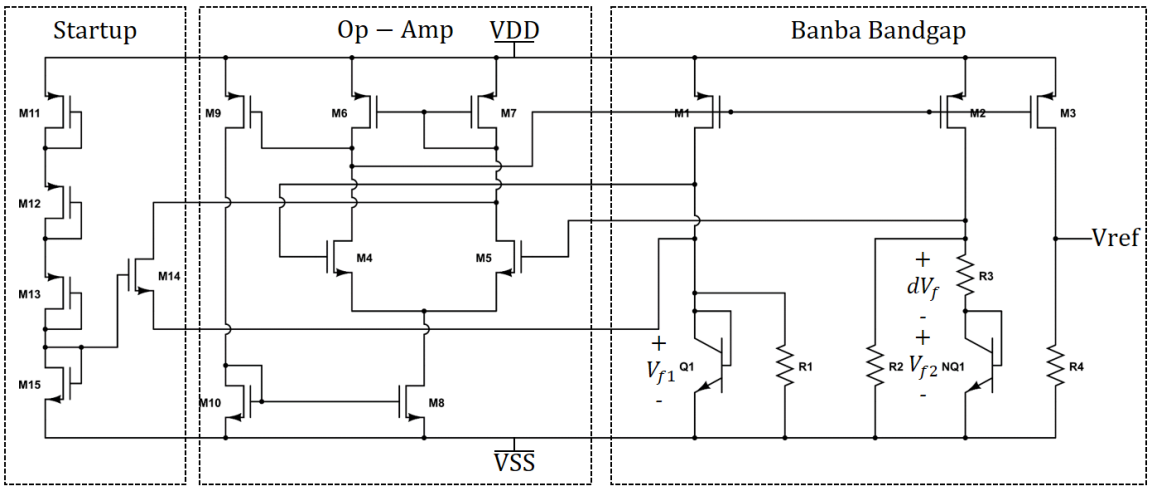

Figure 3.13: A Banba bandgap voltage reference with startup circuitry

The Banba bandgap was chosen for its functionality at low voltages. The Banba bandgap works to remain consistent over varying temperatures by using an op-amp to equalize the sections that are proportional to absolute temperature (PTAT) with sections that are complimentary to absolute temperature (CTAT). Diode connected BJTS (Q1 and NQ1) are CTAT with resistors usually being PTAT.

In Figure 3.13, NQ1 is N BJTs connected together to make the combination of R2, R3, and NQ1 CTAT while the single BJT paired with R1 will be PTAT. N is usual set to 8. These sides enter into opposite terminals of the op-amp so that it can sum them and and set the gates of M1 through M3. Equation (3.10) is then used to determine what the reference voltage is assuming R1 is equal to R2.

$$V_{ref} = R4(\frac{V_{f1}}{R2} + \frac{dV_f}{R3}) \tag{3.10}$$

where V_f1 is the voltage across a diode connected BJT and is inherent to the

technology. dV_f is the difference between the voltages across Q1 and NQ1 and can be determined by

$$dV_f = \frac{kT}{q} ln(N) \tag{3.11}$$

where K is the Boltzmann constant, q is the electron charge,, and T is the current temperature in Kelvin.

The startup circuit supplies a small current when the chip is turned on so that the nodes will not get stuck at either rail and the voltage reference will function. One voltage reference can supply to all spiking neurons in the circuit and can be used to produce multiple different voltages as seen in Figure 3.14.

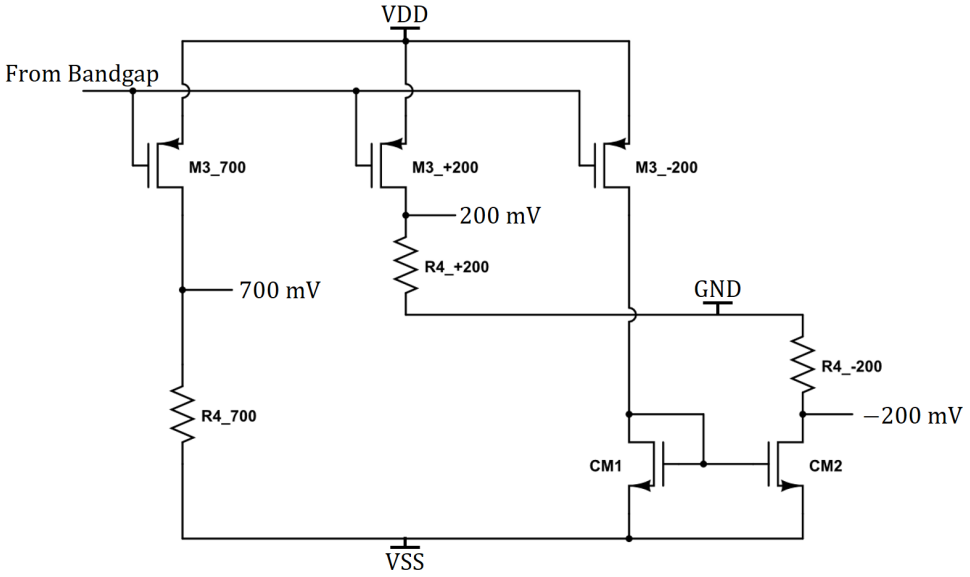

Figure 3.14: Generating different voltages from the Banba bandgap reference

The 700 mV reference voltage is generated off of the VSS rail because it is also used for the current reference which needs the difference between it's supplied voltage and VSS to remain constant when the rail voltages vary. The +/- 200 mV references needed to be referenced off of GND because they are used to drive the inputs of the current amplifier and are restricted by ICMR. To get the negative voltage, the current from $M3_{-200}$ is mirrored and negated with CM1 and CM2.

To supply the reference current for the current amplifier, the circuit at Figure 3.15 is used.

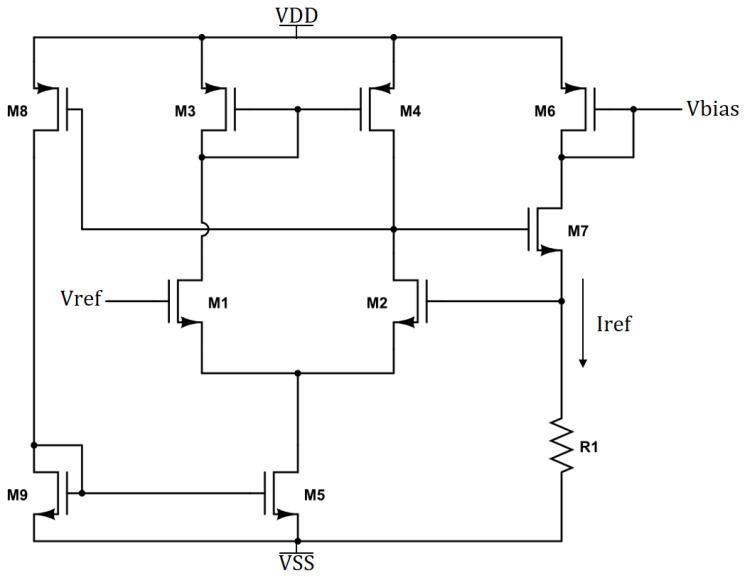

Figure 3.15: A current reference for the current amplifier

The op-amp takes in a reference voltage (700 mV) in the positive terminal and the voltage across R1 into the negative terminal. The feedback loop through M7 allows it to compensate and keep the voltage across R1 equal to V_{ref}. The required current can then be easily set by adjusting the resistance of R1. M6 is used to create a bias point to mirror the current into the op-amp. For each reference current needed, an additional current mirror is added on.

3.8 Memristor Variation

For this thesis, an ideal memristor model was used. General variation in read times, write times, and resistance values can be accounted for during neural network training as was shown in Section 2.6. If the resistance range causes a larger current draw, the crossbar input voltage can be made smaller. if the current is too low, the amplifier can have its gain increased. There is a concern if the memristor threshold voltage changes significantly. In this design, the threshold allows for a 100 mV buffer which

will be partially used by the transmission gates. If a memristor has a wide variance in threshold voltage, the ICMR of the amplifier would need to be increased. In the same vein, a majority of the noise on the signal will be due to the noise of the memristor. If it is too large, higher input voltages can be used to allow for a higher current so the noise would have minimal effect.

Chapter 4

Results

The results are split into three sections: training, spike generation, and sub-circuits. Training is tested on a 2x4 crossbar and shows the capability of the Ziksa crossbar to train in both directions and is tested in nominal only. Spike generation tests a 1x4 crossbar with preset memristor values for it's ability to generate different spiking frequencies in nominal operation. The sub-circuits tests the current amplifier, the comparator, the voltage reference, and the current reference over various corners and in Monte Carlo analysis. This chapter contains mostly tables and few graphs of the results. Details on the sizing and area usage of the circuits can be found in the Appendix.

4.1 Training

To test the training capabilities of the crossbar, A 2x4 crossbar was set up with memristors initialized at varying weights. The test was run for 20 μs The results of training can be seen in Figure 4.1.

In Figure 4.1, the memristor resistance is recorded as a proportional weight of 0 to 1, with 0 being 1 MΩ and 1 being 200 kΩ. This is because when used in a neural network, the lowest resistance will provide the highest current and therefore would be the highest weight. The memristors are first all trained to 0 for 2.25 μs. Due to the higher voltage needed for training, the current required to train all memristors at once

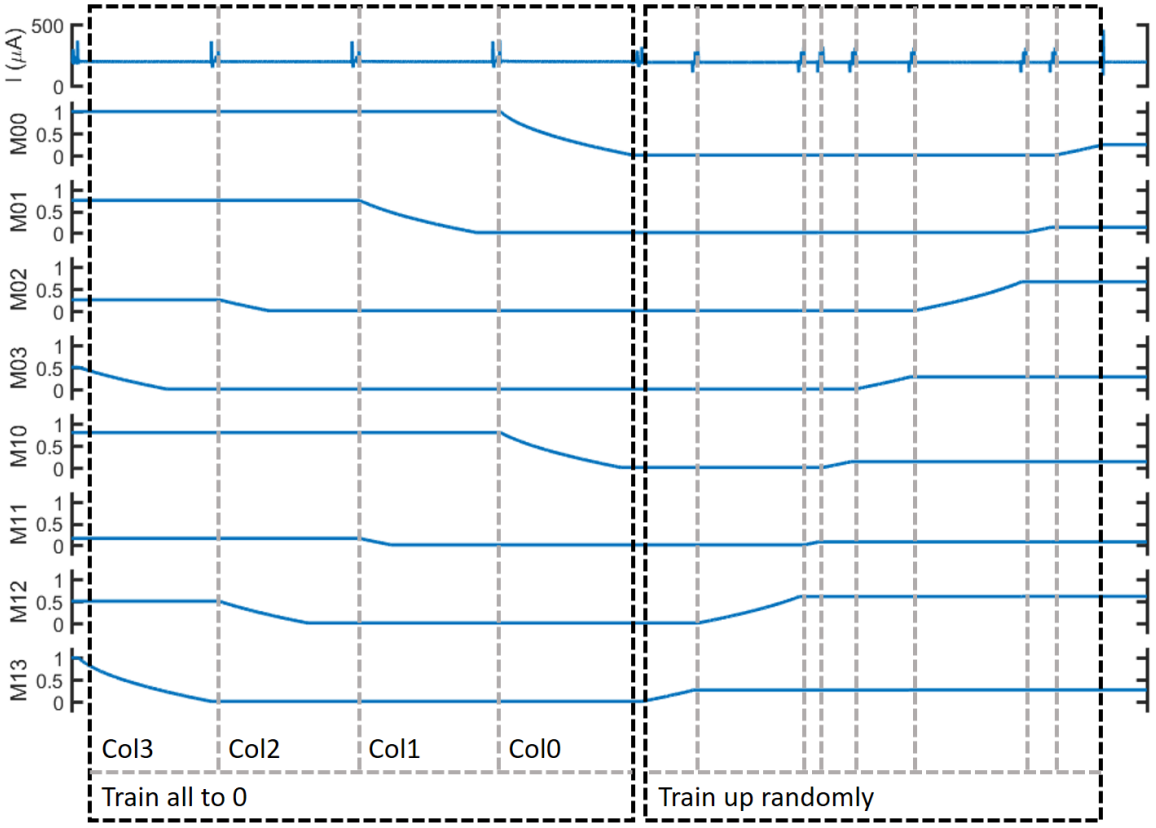

Figure 4.1: Randomly initializing the memristors for use in a neural network

is too much for the current op-amp so they are trained one column at a time. Once all memristors are set to 0, they are individually trained high for random periods of time chosen by an LFSR. This completes the random weight initialization needed in neural networks and shows

During this, the current remains around 190 μA because the largest current needed across a memristor for training is 6 μA. At most, the crossbar could draw 12 μA when training all memristors to 0. This would only be for a short time because as the memristors reach a higher resistance (a relatively lower weight), the current will drop. The rest of the current comes from the external circuitry: transmission gates, voltage reference, current reference, and the current amplifiers. The spiking neuron and it's comparator were not included in this test. With larger crossbars, the number of current amplifiers will increase as there is one needed for every row, but only one

voltage reference and current reference is needed for the entire chip. The spikes in current are from the switching of control signals but have been isolated from the memristors so that the memristors do not accidentally change weight. The current and controls for training a single memristor can be seen in Figure 4.2.

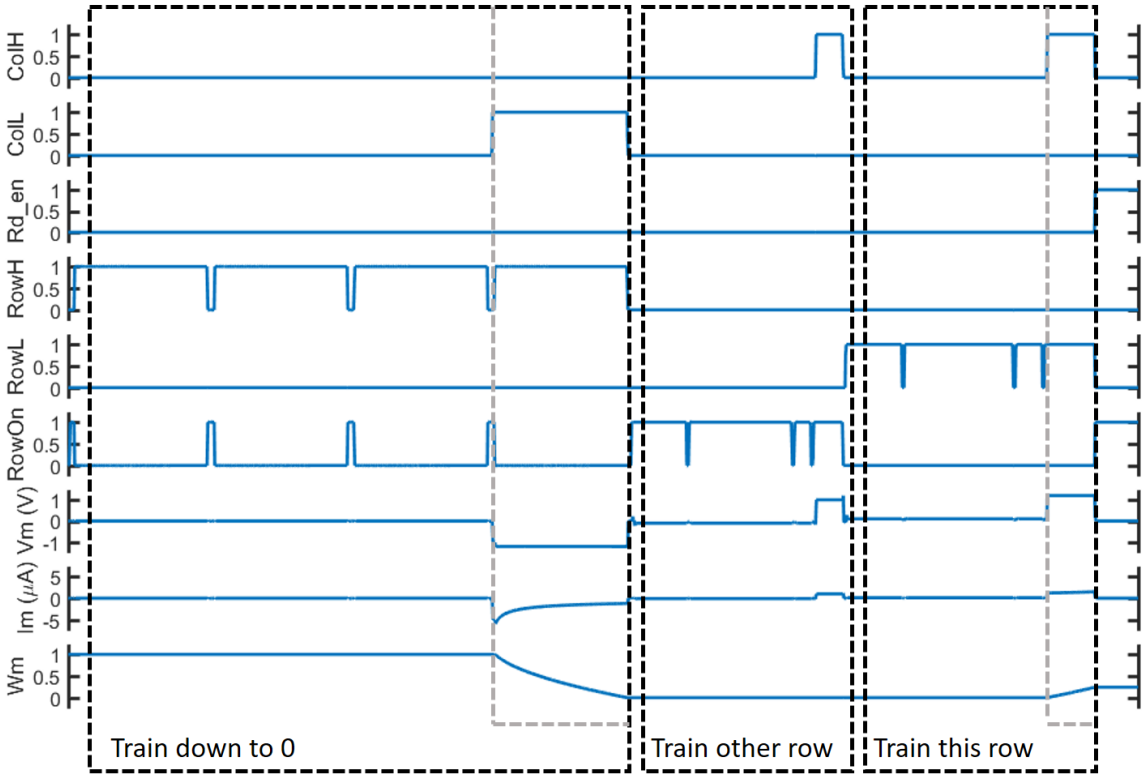

Figure 4.2: The control signals, current across, and resulting weight from training a single memristor.

The top 5 plots are the control signals for the row and column of the memristor being observed. Vm is the voltage across that memristor, Im is the current entering the positive terminal, and Wm is the relative weight of the memristor. The spikes in *RowH*, *RowL*, and *RowOn* occur during the switching of memristors for two reasons: first, to allow for the set-up without training any memristor and second, they are slightly staggered (not visible in this graph) to decrease the current spikes so that the memristors will not train unintentionally. While all memristors are being trained down, the current through the memristor remains around 0 A with spikes that max

at 261.9 nA, until this memristor is trained. Part of the reason for the small current is that the memristors are being trained to 1 MΩ. Once it begins training, the current maxes at 6 μA and decreases as the memristor is trained to a higher resistance. During the random initialization phase, the first half is training memristors in the other rows so the current remains around 90 nA, a higher current due to lower resistances. The current does increase at the end when a memristor in the same column is being trained, but since the row is held to 0 V with the *RowOn* signal, the voltage never exceed 1 V so the memristor remains the same value. During the last phase, the Row is set low so the current hovers around 100 nA while ranging from 1.2 μA to 1.5 μA during training. This demonstrates the low current necessary for memristor crossbars: the majority of power usage comes from the surrounding circuitry.

4.2 Spike Generation

To show the ability of spike generation, five tests were chosen and verified against a Matlab script. The settings can be seen in Table 4.1.

Table 4.1: The frequencies and weights of the spike generation tests

Test	In Frequency (Hz)	Weight ($k\Omega$)	Out
1	20 M, 20 M, 20 M, 20 M	200, 200, 200, 200	23 spikes
2	625 k, 625 k, 625 k, 625 k	1000, 1000, 1000, 1000	None
3	5 M, 2.4 M, 1.25 M, 20 M	500, 200, 750, 400	5 spikes
4	2.5 M, 2.5 M, 2.5 M, 2.5 M	200, 200, 200, 200, 200	None
5	20 M, 20 M, 20 M, 20 M	600, 600, 600, 600	15 spikes

The highest frequency and weight possible was 20 MHz and 200 kΩ. The tests were set up so that test 1 demonstrated the highest values, test 2 the lowest, and tests 3 through 5 showed the variability in between and show the leaking capability of the spiking neuron. The results for test 3 can be seen in Figure 4.3.

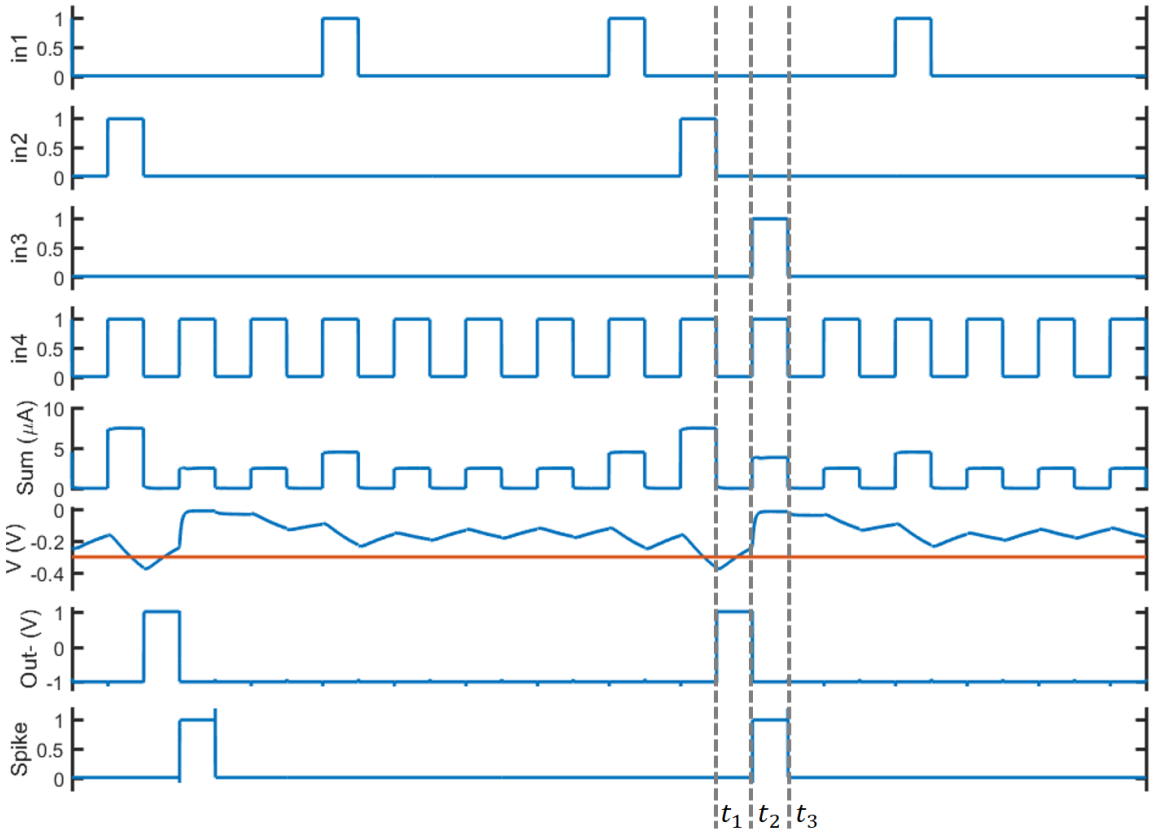

Figure 4.3: The results of the third test from Table 4.1

The first four plots show the input spikes before they pass over their respective memristors. *Sum* shows the summed up currents of these inputs at the input to the current op-amp. V is the voltage in the spiking neuron as it charges, leaks, and spikes. The input spikes were aligned so that there would be moments of 0 A to show the leakage of the spiking neuron. The red line shows the threshold of -0.3 V. *Out-* is the result from the comparator and *Spike* is the final result that would be passed onto further crossbars. At t_1, the comparator trips as the voltage passes the threshold. At the next clock edge, t_2, the SR latch picks up the new value, ANDs it with the clock so that it only be 25 ns wide, and inverters level shift it to 0 V to 1 V. Between t_2 and t_3, a non-shifted *Spike* signals the spiking neuron to drain it's voltage back to 0 V. Finally, at t_3, the spiking neuron can begin charging again. The delay on charging mimics the inhibitory behavior displayed in biological neurons.

Figure 4.4 shows the voltages and spikes of all five tests.

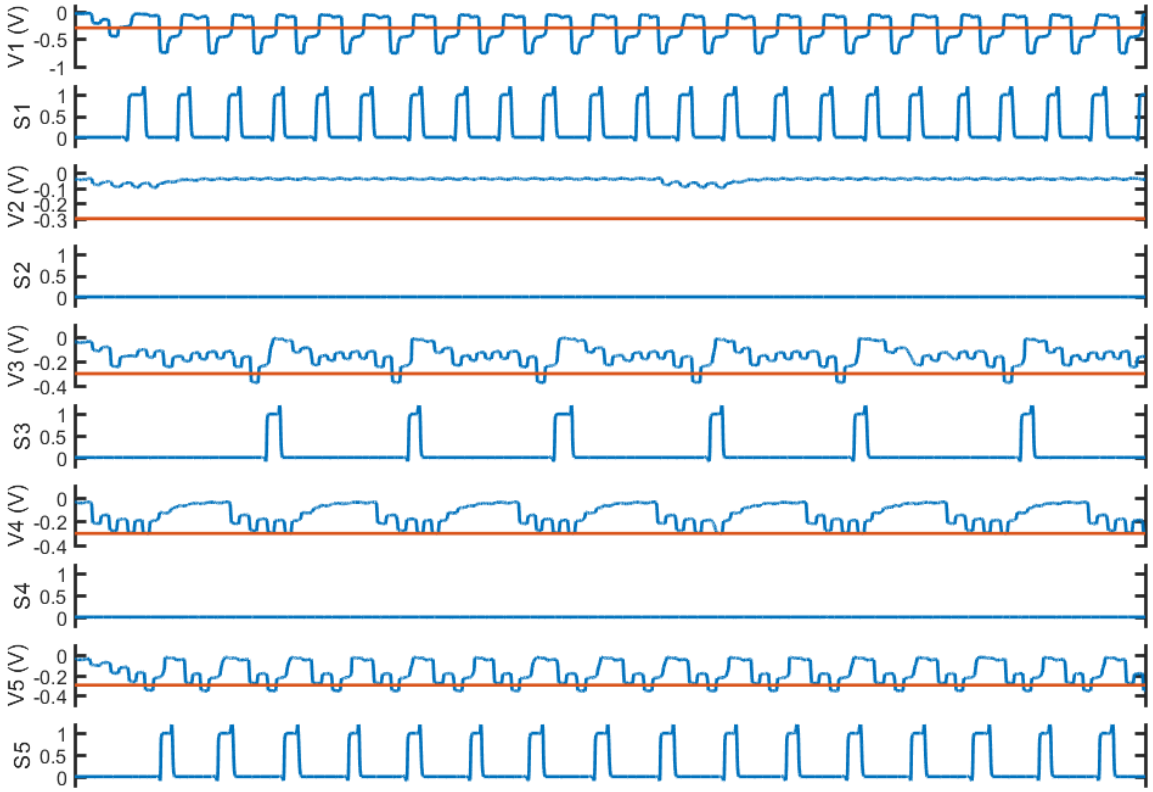

Figure 4.4: The internal voltages and spikes of all five tests from Table 4.1

The first test shows the highest frequency and memristor weight leading to the highest frequency of output spikes. The second test shows a slight increase in voltage, but ultimately returns close to 0 V. The reason the voltage does not reach a true 0 V is due to a slight current that will still be produced with a 0 A input. This is due to non-idealities in the current op-amp and is discussed in Section 4.3.1. The current does not cause a major issue because the leaking constant of the spiking neuron will keep low currents from charging up the neuron to spike. The third test was discussed in detail above. Test 4 reaches a voltage of -0.3 V, but the comparator has a limit for how close it must be to regularly fire, so it does not spike. This also shows the voltage discharging slowly. The rate of discharge is entirely controlled by the timing constant of the spiking neuron. The final test simply shows another possible frequency being generated. The rate at which the neuron fires can be altered by changing the input

currents, memristor range, threshold voltage of the spiking neuron, or the timing constant of the spiking neuron.

4.3 Analog Circuits

The next section is a detailed analysis of the current amplifier, comparator, voltage reference, and current reference. All were tested over PVT (Process, Voltage, Temperature) corners with process covering five corners of device variation (fast-fast, fast-slow, slow-slow, slow-fast, typical-typical), voltage changing setting VDD to typical and +/- 0.2 V, and temperature tested at 0 °C, 27 °C, and 100 °C. They were also tested over 1000 runs of Monte Carlo simulation at nominal voltage and temperature. Monte Carlo varies the variables used to describe individual transistor behavior using a Gaussian distribution. The variation highlights the issues that might arise due to transistor mismatch.

4.3.1 Current Amplifier

The current amplifier was tested with a load resistance of 100 kΩ and a capacitance of 500 fF, to match the spiking neuron. The resulting nominal, PVT, and Monte Carlo results can be seen in Table 4.2.

Table 4.2: PVT and Monte Carlo simulation results for the current amplifier

	Specs	**Nom**	**PVT**	**MC**
Power(uW)	minimize	78.47	[63.07, 96.37]	[74.9, 83.75]
Gain (A/A)	[-0.95,-1.05]	1.01	[1.004, 1.049]	[1.003, 1.022]
Phase Margin (°)	$\geq$ 60	75.66	[61.53, 88.65]	[64.79, 96.62]
Bandwidth (MHz)	$\geq$ 20	87.5	[43.18, 100.2]	[56.03, 72.77]
ICMR Max (mV)	$\geq$ 200	364.2	[249.4, 476.2]	[329.4, 409.2]
ICMR Min (mV)	$\leq$ -200	-355.1	[-510.1, -205.1]	[-385.1, -320.1]
Input Offset (mV)	$\leq$ 10	514.9u	[372.9u, 880.9u]	[2.087u, 4.711m]
Intersect (nA)	[-1uA, 1uA]	-231.1	[-388.1, -127.8]	[-423.1, 13.15]
Curr In Offset (nA)	$\leq$ 100	7.952	[4.297, 41.76]	[0.2463, 88.65]
Curr Out Diff (uA)	$\leq$ 1.5	1.101	[0.891, 1.289]	[1.057, 1.209]

Most specifications tested were standard for an amplifier. Power was minimized but had no hard requirement. The only required gain was the current gain of -1 A/A with a 5% variation. Phase margin was limited to 60° and was measured over the feedback loop of the current amplifier which ran from the differential inputs to inbetween M10 and M12 of Figure 3.6. The resulting bode plot can be seen in Figure 4.5.

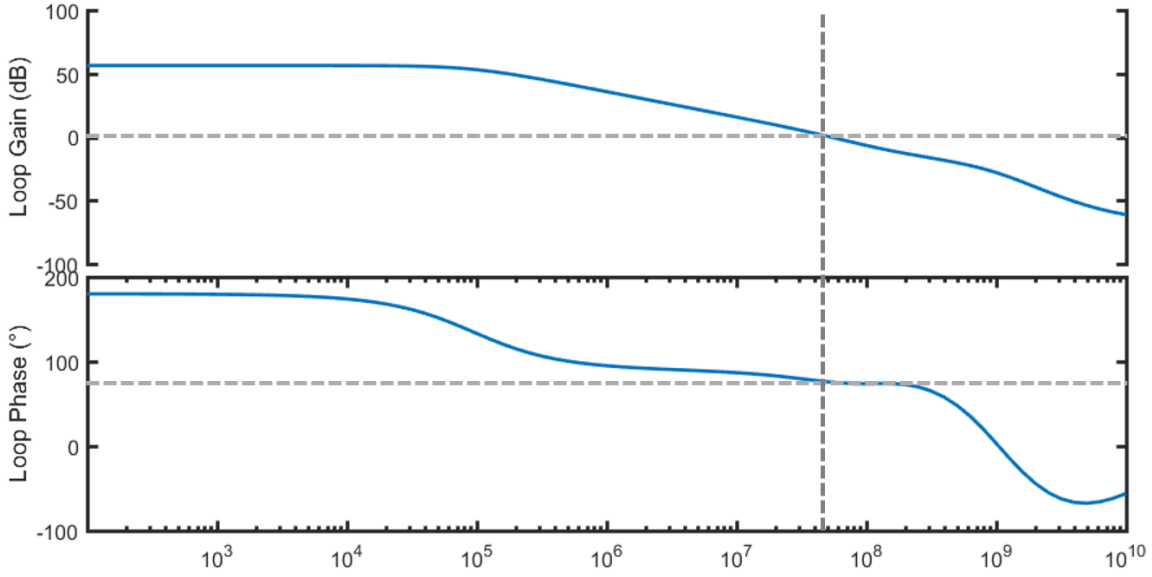

Figure 4.5: Bode plot of the loop gain and phase during the nominal run

The dark gray line shows where the gain is 1 dB and the phase margin is measured. For nominal simulation, this was 75.66°. The exact bandwidth of the loop does not matter because it is connected in unity gain configuration. Instead bandwidth was measured over the full current amplifier at the current gain of -1 A/A. It was limited to 20 MHz to match the spiking inputs. A square wave will have higher harmonics, but the RC pair of the spiking neuron works as a low pass filter so only a bandwidth of 20MHz is needed. ICMR (Input Common Mode Range) had a very strict requirement of -0.2 V to 0.2 V so that it would be able to train the memristors. A higher range would be preferred so that the memristors could be trained faster, but that would have required a larger op-amp output stage such as the Monticelli output stage [31]. ICMR

was measured by connecting the op-amp in unity gain configuration and sweeping the positive input from -1 V to 1 V. The voltage was measured at the output and the double derivative used to determine the maximum and minimum voltages at which the output would follow the sweeping input. The input offset of the amplifier was measured as the difference between the input voltages and the histogram for the Monte Carlo simulation is contained in Figure 4.6.

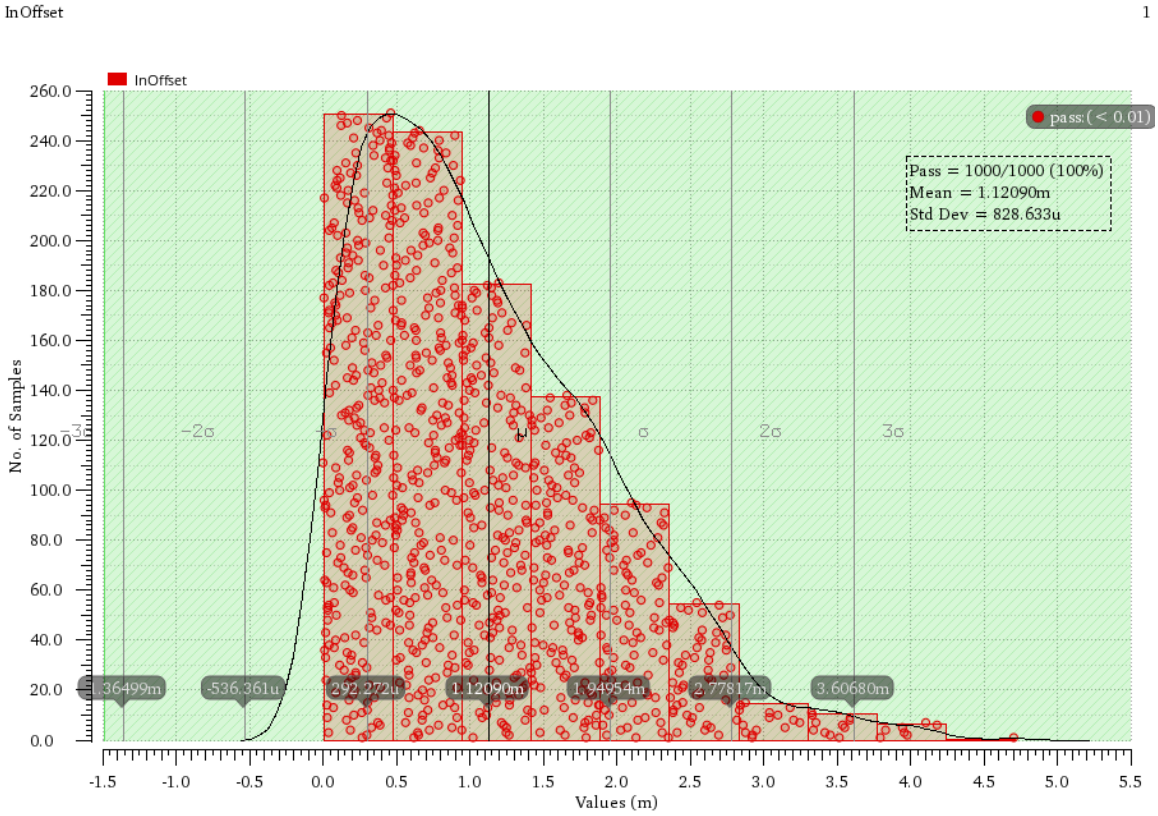

Figure 4.6: Histogram of input offset over Monte Carlo simulations

Input offset was limited to 10 mV for accurate training voltages and input current. Training voltages only have 100 mV of space above the memristor threshold voltage so an inaccurate input voltage, coupled with transmission gates passing an inaccurate VDD and VSS, could prevent training. With input spikes at 1 V, a large offset would lead to a significant change in input current. Over Monte Carlo, the mean offset was 1.1206 mV with a standard deviation of 828 μV.

The last three specifications are unique to the transfer function of the current

amplifier. Figure 4.7 shows the graphs that these results were pulled from.

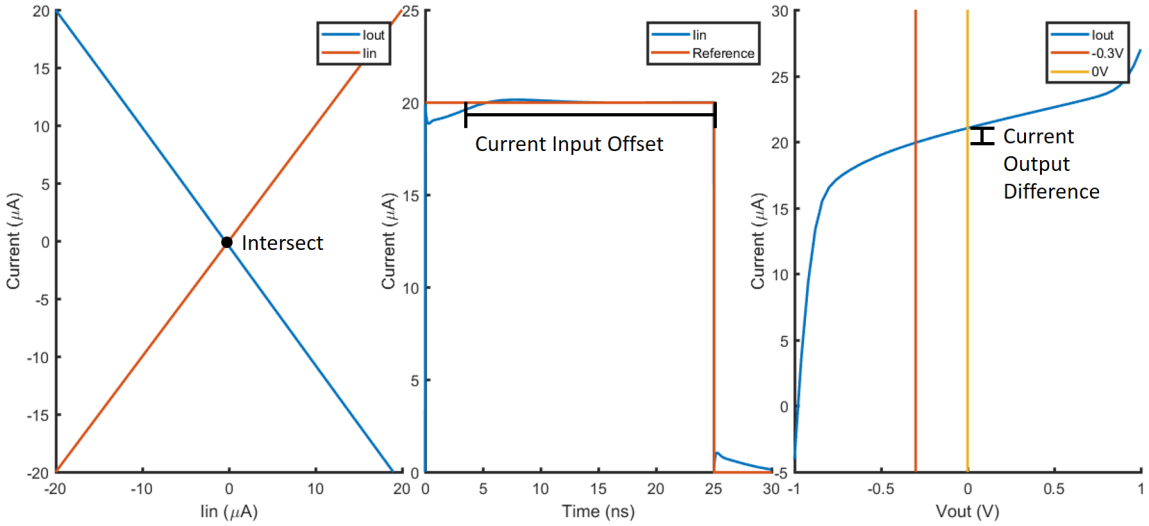

Figure 4.7: Graphs used to determine intersect, current input offset, and current output difference

The intersect was determined by sweeping the input current and finding the point at which the input current intersects the output current. This measures the offset between input and output current. Ideally, they should intersect at 0 A, but a small variation due to offset and gain error will not cause an issue because any small current generated will not effect the spiking neuron due to the leaking constant. The current input offset tests that the amplifier is able to source enough current for operation. The maximum current needed is 20 μA so the offset is how far off the current is from 20 μA. This was averaged from 3 ns to 25 ns to avoid the beginning undershoot. The effect of the undershoot is already measured in the slew rate for data being passed through the op-amp. For training, since the signal will always be at least 50 ns long, the first 3 ns are not as important as verifying that the remaining signal is close to the required current. The Monte Carlo simulation histogram can be seen in Figure 4.8.

The mean offset was 21.1179 nA and the standard deviation was 15.5562 nA. In relation to the 20 μA signal, this offset will cause minimal effect.

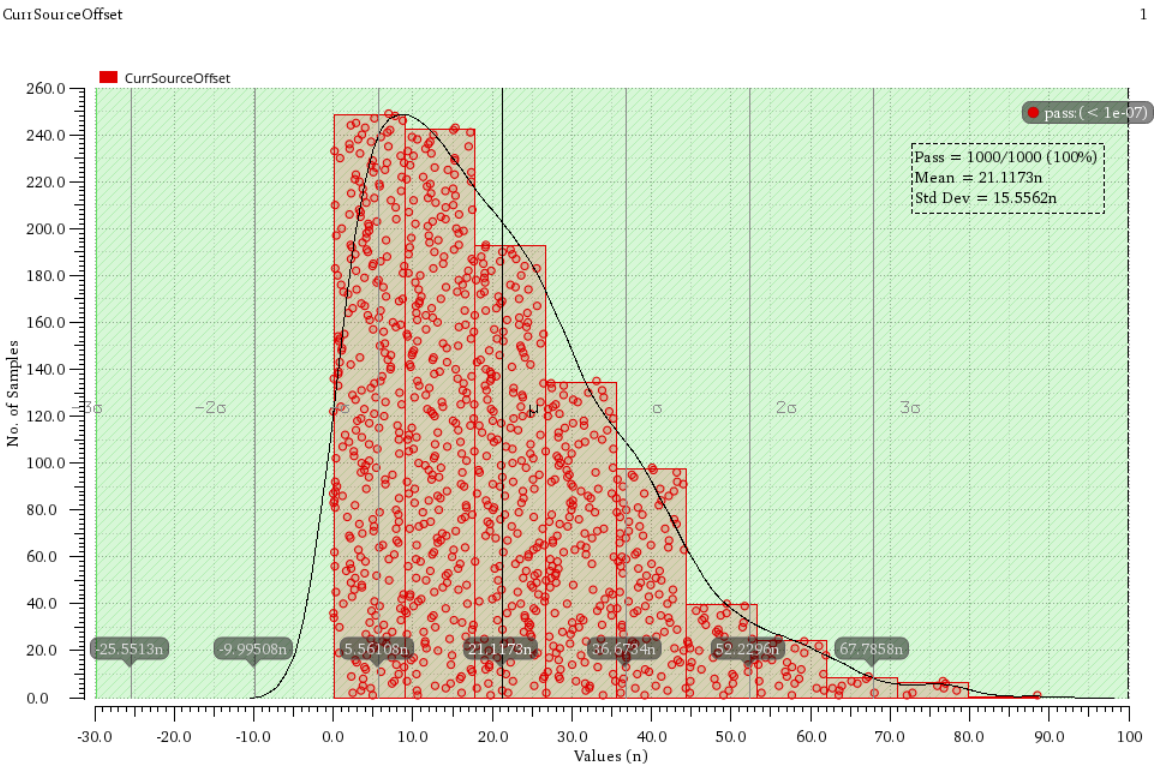

Figure 4.8: Histogram of current input offset over Monte Carlo simulations

The current output difference was measured in Figure 4.7 by pushing a constant 20μA into the amplifier and sweeping the output voltage. The output should only move between -0.3 V and 0 V due to the spiking neuron behavior, so the requirement was that there is a max of 500 nA change in output current over each 0.1 V of output voltage change. This essentially is measuring the output impedance of the current amplifier. For a 500 nA over 0.1 V, that leads to a minimum output resistance of 200 kΩ.

Of the op-amp specifications, the largest difficulty was balancing ICMR and the current output difference. ICMR could be lowered by making the M15/M16 pair larger, but this significantly increased the current difference. The internal current mirrors, M9/M10 and M11/M12, could be made wide and short to improve ICMR, but this also hurt the output difference. A careful balance of sizing these brought values within specification. Other topologies could've provided better ICMR, but

they would've required a much larger amplifier. Seeing as one amplifier is required for every neuron, keeping the size minimal is paramount.

4.3.2 Comparator

The comparator was also tested over PVT and 1000 runs of Monte Carlo at nominal. The tabular results can be seen in Table 4.3 while the graphical results are in the Appendix.

Table 4.3: The resulting PVT and Monte Carlo simulation results for the comparator

	Spec	**Nominal**	**PVT**	**Monte Carlo**
Power (μW)	minimize	145.6	[105.6, 196.9]	[145.9, 167.7]
Input Drive (mV)	≤ 20	-	-	-
Offset Voltage (mV)	≤ 10	47.68n	[-47.68n, 47.68n]	[-9.194, 8.828]
Kickback (mV)	≤ 10	7.375	[6.720, 8.119]	[7.7116, 7.594]
Hysteresis (mV)	≤ 5	1.503	[0.361, 3.377]	[0.412, 2.577]

The input drive voltage is the minimum voltage difference the comparator can detect and was tested by verifying the comparator correctly computed eight comparisons during transient operation. The offset voltage is the voltage difference between the two inputs and was measured with a veriloga script [32]. Offset is negligable during PVT simulations because the main cause of offset is mismatch between the input transistors. Mismatch between the same types of transistors is only simulated during Monte Carlo.

Kickback is how much the rising clock edge causes a voltage spike on the inputs. A large kickback would cause changes in the current charge of the spiking neuron. Finally, hysteresis is the comparator having memory of the previous result which can influence the next comparison. To check for hysterisis, the internal nodes from the left and right side were compared directly after the clock rose high. As long as the difference was less than 5 mV, any memory effect was considered negligible.

Due to the voltage reference variation, the comparator was also tested at +/- 0.075V off of 700 mV. The results can be seen in Table 4.4.

Table 4.4: The PVT results for +/- 75 mV off of a voltage reference of 700 mV

	Spec	**Vref = 625mV**	**Vref = 775mV**
Power (μW)	minimize	[94.58, 190.9]	[111.9, 204.9]
Input Drive (mV)	≤ 20	-	-
Offset Voltage (mV)	≤ 10	[-47.68n, 47.68n]	[-47.68n, 47.68n]
Kickback (mV)	≤ 10	[6.392, 7.945]	[6.821, 8.247]
Hysteresis (mV)	≤ 5	[0.210, 2.079]	[623.7, 4.872]

The results were very similar at 700mV or +/- 75 mV minus a slight increase in hysteresis. The main problems with sizing the comparator was when testing it at the low voltage corners where it would often miss comparisons due to needing more current and lower RC constants at the nodes. First, the latching transistors were kept small to reduce their RC constant. The bottom transistor of the differential amplifier was made wide to increase the current it drew. The input transistors were also made wide to increase the gain, but this increased the input capacitance which would increase the kickback. Offset is also benefited by larger input transistors, but these lead to a high power draw. A careful balance was achieved and the resulting sizes can be seen in Table A.3.

4.3.3 Voltage and Current Reference

For both the voltage and current reference, their main requirement was accuracy measured with PPM (Parts Per Million) which can be calculated with

$$PPM = \frac{Max - Min}{100 * average} * 10^6. \tag{4.1}$$

PPM is measured over sweeping the temperature and independent of the reference voltage or current. Area is not a major concern because only one voltage and current reference is needed for the entire chip. Since temperature was swept for PPM, both reference circuits were tested over only PV and Monte Carlo at nominal. The results can be seen in Table 4.5 and 4.6

Table 4.5: PV and Monte Carlo simulation results for the voltage reference.

		Spec	Nom	PV	Monte Carlo
	Power (μW)	minimize	67.55	[54.47, 107]	[71.42, 77.89]
700 mV	**Vref (mV)**	700	700.1	[678.5, 700.7]	[650.2, 712.1]
	PPM (PPM/°C)	$\leq$ 200	103	[45.8, 131.3]	[27.32, 197.3]
	Range (mV)	-	7.213	[3.11, 9.044]	[1.809, 13.93]
200 mV	**Vref (mV)**	200	200	[196.0, 207.4]	[189.7, 211.1]
	PPM (PPM/°C)	$\leq$ 200	66.72	[10.46, 174.1]	[4.786, 194.2]
	Range (mV)	-	1.334	[0.213, 3.494]	[0.099, 3.685]
-200 mV	**Vref (mV)**	-200	-200	[-206.1, -195.7]	[-211.6, -189.8]
	PPM (PPM/°C)	$\leq$ 200	19.00	[5.624, 39.94]	[2.827, 129.8]
	Range (mV)	-	0.380	[0.110, 0.810]	[0.055, 2.737]

The 700 mV reference was referenced off the VSS rail because it would also be used for the current reference. The 200 mV and -200 mV references were referenced off the GND rail because they would be used for the op-amp positive input whose ICMR is based off the GND rail as well. Reaching the required PPM was mostly accomplished with using large BJTs and balancing the ratio of the resistors. While all references shared the majority of the Banba bandgap, they each had their own resistor to set their individual voltages. This lead to the major difference in variation between the 700 mV reference and the +/-200 mV references. The smaller references had similarly sized resistors while the 700 mV needed a much larger resistor leading to a larger PTAT contribution. The bandgap could then be tuned to either the higher PTAT of the 700 mV or the smaller PTAT of the +/-200 mV. Since the +/-200 mV reference was limited by the training voltages and ICMR, is was more important that the smaller references had minimal variation than the 700 mV. In particular, the -200 mV reference needed to satisfy the minimum ICMR requirement of -205.1 mV from Table 4.2. This reference slightly misses the mark with a -210.9 mV Monte Carlo result, but a few mV into the triode region should only decrease the op amp gain slightly. A future improvement would be to push the op-amps ICMR a few mV further to fix this issue. The current reference results can be seen in Table 4.6.

The current reference was easy to adjust due to it having a single resistor that

Table 4.6: The PV and Monte Carlo simulation results for the current reference.

	Spec	**Nominal**	**PV**	**Monte Carlo**
Power (μW)	minimize	53.19	[34.73, 83.48]	[44.07, 80.69]
Iref (μA)	10	10	[8.566, 13.26]	[8.26, 13.01]
PPM (PPM/°C)	≤ 200	28.71	[10.69, 85.03]	[11.47, 102.4]
Range (nA)	-	28.71	[9.669, 72.84]	[11.47, 116]

controls the output current. The amplifier was sized up to improve accuracy which added to the large power draw. Since there is only a single current reference, the power draw will be minimal in comparison to the full design.

4.3.4 Area and Power

Though there is no layout, the area for most circuitry can be grossly estimated using

$$\begin{aligned} A_{est}(\mu m^2) = & \sum_{transistors} W(L + 0.28\mu m) + \\ & \sum_{resistors} segments(W \times L) + \\ & \sum_{capacitors} multiplier(W \times L). \end{aligned} \tag{4.2}$$

The layout style, dummy transistors, and shielding will cause a large difference in the actual sizings, but the equation is sufficient to ballpark the largest portions of this circuitry. The $0.28\mu m$ comes from the Pcells of the transistor node and is the additional area required for the source and drain of the transistor. Since it is common to share sources and drains, the area during layout will be smaller. The device size break-down and percentages can be found in Tables A.2 through A.5 in the Appendix. The final areas can be seen in Table 4.7.

Table 4.7: Approximate areas of the circuitry for a 2x4 crossbar

	Area(μm^2)	**Percentage (%)**
ZTC	15.6	0.138
Current Amplifier	953	8.48
Comparator	26.9	0.239
Spiking Neuron Circuitry	3,120	27.8
Voltage Reference	6,800	60.6
Current Reference	312	27.8
Crossbar	0.065	0.0006
Total Area	11,227.5	

This does not include the control unit circuitry. ZTC is all the circuitry required for the training units minus the amplifier. The spiking neuron circuitry is the neuron, SR latch, and inverters but not the comparator. Since this was not designed with a specific crossbar in mind, the memristor crossbar size can be estimated based on each memristor (and the area between memristors) taking up $4F^2$ where F is the minimum features size of the layout technique [33]. The resulting size is negligible compared to the peripheral circuitry. For all components, the dominating factor in size is their resistors and capacitors. The voltage reference has four resistors, the largest at 270 kΩ, and the current reference has one resistor. The current amplifier has a small resistor, but a 243.8fF capacitor that takes up 47% of its area. The comparator and ZTC, minus the op-amp, are only transistors so are negligible size. The spiking neuron circuitry contains a very large resistor and capacitor. While smaller devices are available, they have large parasitic resistances and capacitances that prevent them from following the expected charging and discharging curves. As the number of neurons increases, the dominating circuitry changes as can be seen in Figure 4.9.

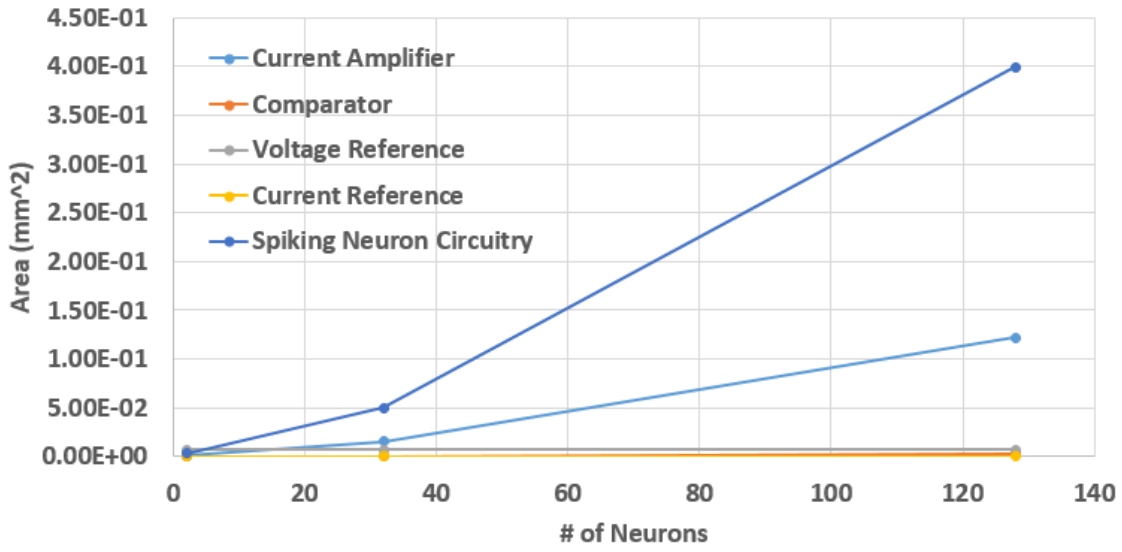

Figure 4.9: Area usage of different circuits over increasing numbers of neurons

For a neural network to achieve high accuracy on large problems, it needs hundreds of neurons. The area usage was extrapolated from a 2x4 (2 neurons) to a 32x32 crossbar (32 neurons) and then to four 32x32 crossbars (128 neurons). With the size increase, the dominating circuitry is the spiking neuron due to it's resistor and capacitor. The focus of this thesis was on the training circuitry so a simple spiking neuron was chosen to prove the training unit functionality in a full circuit. More complex, but smaller, spiking neurons exist such as in [34] with an area of 80.406 μm^2 per neuron. In this case, the current amplifier would dominate as size increases. The main cause of area usage, the compensation capacitor, is necessary to keep the phase margin high enough to prevent oscillations. Careful resizing could potentially reduce it's size, but the main factor in its size will depend on the final layout.

A table of the power usage for a 1x4 crossbar can be seen in Figure 4.8.

Table 4.8: Power usage for a 1x4 crossbar

	Power (μW)	**Percentage** (%)
Current Amplifier	85.16	36.74
Comparator	2.339	1.01
Voltage Reference	67.69	29.2019
Current Reference	52.69	22.7308
Crossbar	10.00	4.314
Extra	23.921	6.005
Total	231.8	

These power measurements were taken over the 1x4 transient tests because the included all the components. It can be seen that the current amplifier is the dominant usage of power at 36.74 μW. The comparator's power usage is much smaller than what was reported in Table 4.3 because the majority of power is used when the clock switches and in these simulations, the clock switches once every 25 ns. For simplicity, the power usage of all transmission gates, inverters, and the SR latch were included in the extra category. If extrapolating to a 32x32 crossbar, the total power would be 3.685 mW with the amplifier using 73.9% of the power.

Chapter 5

Final Remarks

The work presented is a stepping stone for larger networks and circuitry and a significant improvement on the Ziksa training circuitry which was previously only implemented with ideal amplifiers. High-level simulations show it training a 2x4 memristor crossbar with minimal wasted current and show it functioning within the context of a spiking neuron. Each analog component passes specifications over process, supply voltage, and temperature corners along with 1000 runs of Monte Carlo simulation. While not tested on a data set, it shows the behavior necessary to function in a full neural network.

5.1 Future Works

While functional, many improvements can be made to the circuitry. The main issue discussed was the massive size of the spiking neuron due to it's resistor and capacitor. Other designs exist that use only a capacitor and the inherent resistance of transistors. Also, a smaller capacitor could be used if the current is reduced. Of course a smaller current will mean noise has a larger impact on the signal. For the amplifier, it was briefly mentioned that a different output stage could improve the ICMR of the amplifier. A higher ICMR would open the circuit to work with less accurate memristors and analog circuitry which can help reduce the size. While the different output stage would be larger, the amplifier is dominated by the compensation capacitor and

resistor so the additional transistors may not cause a major issue. Another simple improvement would be to change the output stage to a cascoded output which would decrease the variation in output current as the output voltage varies.

The next major step is to use the Ziksa training circuitry in a full neural network. This would involve larger crossbars which, depending on the memristor resistance ranges, may necessitate slight changes in the amplifier. More likely if the current draw is too high, simply reducing the input signal voltage with level shifting will reduce the read current. Since memristors are only trained one at a time, the amount of current needed during training will not change with larger crossbars. It should also be implemented with models of an existing memristor with process variation. Process variation will not break the overall design, but specifications such as the ICMR and the training voltages may need to change if the memristor threshold voltage varies too much.

While the area estimations are a good starting point for the area consumption, a layout is needed to truly determine if this design is an improvement over other similar designs. Since the amplifier depends heavily on current mirrors, matching will be a major issue in the final accuracy. To improve the matching, dummy transistors and interdigitizing will be necessary which can greatly increase the area used. Since these amplifiers are used repeatedly, some accuracy will need to be sacrificed to keep the area from significantly increasing. Similarly, the control unit needs to be fully synthesized. While it should be small in comparison to the analog circuits, there is a large number of control signals that need to be carefully laid out.

In simulation, a small test-case can be used to verify functionality and accuracy of learning, but simulating large networks will only be possible with a Matlab approximation due to Cadence simulation run-time. To see the full performance, the circuit needs to be physically built.

Bibliography

[1] C. Merkel, "Thermal profiling in cmos/memristor hybrid architectures," Ph.D. dissertation, 2011.

[2] A. M. Zyarah, N. Soures, L. Hays, R. B. Jacobs-Gedrim, S. Agarwal, M. Marinella, and D. Kudithipudi, "Ziksa: On-chip learning accelerator with memristor crossbars for multilevel neural networks," in *Circuits and Systems (ISCAS), 2017 IEEE International Symposium on*. IEEE, 2017, pp. 1–4.

[3] N. Soures, L. Hays, E. Bohannon, A. M. Zyarah, and D. Kudithipudi, "On-device stdp and synaptic normalization for neuromemristive spiking neural network."

[4] "What gives at dropout? low dropout regulator performance near dropout," Sep 2015. [Online]. Available: https://www.ecnmag.com/article/2010/12/what-gives-dropout-low-dropout-regulator-performance-near-dropout

[5] T. Tang, L. Xia, B. Li, R. Luo, Y. Chen, Y. Wang, and H. Yang, "Spiking neural network with rram: Can we use it for real-world application?" in *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition*. EDA Consortium, 2015, pp. 860–865.

[6] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain." *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[7] K.-i. Funahashi and Y. Nakamura, "Approximation of dynamical systems by continuous time recurrent neural networks," *Neural networks*, vol. 6, no. 6, pp. 801–806, 1993.

[8] M. LukošEvičIus and H. Jaeger, "Reservoir computing approaches to recurrent neural network training," *Computer Science Review*, vol. 3, no. 3, pp. 127–149, 2009.

[9] D. Kudithipudi, Q. Saleh, C. Merkel, J. Thesing, and B. Wysocki, "Design and analysis of a neuromemristive reservoir computing architecture for biosignal processing," *Frontiers in neuroscience*, vol. 9, 2015.

[10] L. Chua, "Memristor-the missing circuit element," *IEEE Transactions on circuit theory*, vol. 18, no. 5, pp. 507–519, 1971.

[11] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The missing memristor found," *nature*, vol. 453, no. 7191, pp. 80–83, 2008.

[12] A. Hellemans, "Six-state memristor opens door to weird computing [news]," *IEEE Spectrum*, vol. 12, no. 51, p. 18, 2014.

[13] T. Skotnicki, J. A. Hutchby, T.-J. King, H.-S. Wong, and F. Boeuf, "The end of cmos scaling: toward the introduction of new materials and structural changes to improve mosfet performance," *IEEE Circuits and Devices Magazine*, vol. 21, no. 1, pp. 16–26, 2005.

[14] A. N. Burkitt, "A review of the integrate-and-fire neuron model: I. homogeneous synaptic input," *Biological cybernetics*, vol. 95, no. 1, pp. 1–19, 2006.

[15] B. Schrauwen, D. Verstraeten, and J. Van Campenhout, "An overview of reservoir computing: theory, applications and implementations," in *Proceedings of the 15th European Symposium on Artificial Neural Networks. p. 471-482 2007*, 2007, pp. 471–482.

[16] H. Jaeger, "Echo state network," vol. 2, no. 9, p. 2330, 2007, revision 151757.

[17] H. Abunahla, D. Homouz, Y. Halawani, and B. Mohammad, "Modeling and device parameter design to improve reset time in binary-oxide memristors," *Applied Physics A*, vol. 117, no. 3, pp. 1019–1023, 2014.

[18] M. Noman, W. Jiang, P. A. Salvador, M. Skowronski, and J. A. Bain, "Computational investigations into the operating window for memristive devices based on homogeneous ionic motion," *Applied Physics A: Materials Science & Processing*, vol. 102, no. 4, pp. 877–883, 2011.

[19] G. S. Snider, "Architecture and methods for computing with reconfigurable resistor crossbars," Apr. 10 2007, uS Patent 7,203,789.

[20] K.-H. Kim, S. Gaba, D. Wheeler, J. M. Cruz-Albrecht, T. Hussain, N. Srinivasa, and W. Lu, "A functional hybrid memristor crossbar-array/cmos system for data storage and neuromorphic applications," *Nano letters*, vol. 12, no. 1, pp. 389–395, 2011.

[21] M. A. Zidan, H. A. H. Fahmy, M. M. Hussain, and K. N. Salama, "Memristor-based memory: The sneak paths problem and solutions," *Microelectronics Journal*, vol. 44, no. 2, pp. 176–183, 2013.

[22] Y. Cassuto, S. Kvatinsky, and E. Yaakobi, "Sneak-path constraints in memristor crossbar arrays," in *Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on*. IEEE, 2013, pp. 156–160.

[23] S. H. Jo, T. Chang, I. Ebong, B. B. Bhadviya, P. Mazumder, and W. Lu, "Nanoscale memristor device as synapse in neuromorphic systems," *Nano letters*, vol. 10, no. 4, pp. 1297–1301, 2010.

[24] D. Soudry, D. Di Castro, A. Gal, A. Kolodny, and S. Kvatinsky, "Memristor-based multilayer neural networks with online gradient descent training," *IEEE transactions on neural networks and learning systems*, vol. 26, no. 10, pp. 2408–2421, 2015.

[25] R. Hasan, T. M. Taha, and C. Yakopcic, "On-chip training of memristor crossbar based multi-layer neural networks," *Microelectronics Journal*, vol. 66, pp. 31–40, 2017.

[26] N. Soures, L. Hays, and D. Kudithipudi, "Robustness of a memristor based liquid state machine," in *Neural Networks (IJCNN), 2017 International Joint Conference on*. IEEE, 2017, pp. 2414–2420.

[27] H.-Y. Huang, B.-R. Wang, and J.-C. Liu, "High-gain and high-bandwidth rail-to-rail operational amplifier with slew rate boost circuit," in *Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium on*. IEEE, 2006, pp. 4–pp.

[28] J. M. Miller, *Dependence of the input impedance of a three-electrode vacuum tube upon the load in the plate circuit*. Government Printing Office, 1919, vol. 351.

[29] H. Jeon and Y.-B. Kim, "A low-offset high-speed double-tail dual-rail dynamic latched comparator," in *Proceedings of the 20th Symposium on Great Lakes Symposium on VLSI*, ser. GLSVLSI '10. New York, NY, USA: ACM, 2010, pp. 45–48. [Online]. Available: http://doi.acm.org/10.1145/1785481.1785493

[30] H. Banba, H. Shiga, A. Umezawa, T. Miyaba, T. Tanzawa, S. Atsumi, and K. Sakui, "A cmos bandgap reference circuit with sub-1-v operation," *IEEE Journal of Solid-State Circuits*, vol. 34, no. 5, pp. 670–674, 1999.

[31] D. M. Monticelli, "A quad cmos single-supply op amp with rail-to-rail output swing," *IEEE Journal of Solid-State Circuits*, vol. 21, no. 6, pp. 1026–1034, 1986.

[32] D. Levski, "Offset measurement of latched comparators." [Online]. Available: https://transistorized.net/post/stdal/post3.htm

[33] K. Eshraghian, K.-R. Cho, O. Kavehei, S.-K. Kang, D. Abbott, and S.-M. S. Kang, "Memristor mos content addressable memory (mcam): Hybrid architecture for future high performance search engines," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 19, no. 8, pp. 1407–1417, 2011.

[34] G. Indiveri, E. Chicca, and R. Douglas, "A vlsi array of low-power spiking neurons and bistable synapses with spike-timing dependent plasticity," *IEEE transactions on neural networks*, vol. 17, no. 1, pp. 211–221, 2006.

Appendices

Appendix A

Device Sizes

Table A.1: Resistor and capacitor values for all circuitry

Circuit	**Device**	**Value**
Current Amplifier	R_z	3.25 kΩ
	C_c	243.8 fF
Voltage Reference	R1	275 kΩ
	R2	275 kΩ
	R3	18.3 kΩ
	R4 (700 mV)	140.29 kΩ
	R4 (200 mV)	46.44 kΩ
	R4 (-200 mV)	38.34 kΩ
Current Reference	R1	73.13 kΩ
Spiking Neuron Circuitry	R	100 kΩ
	C	500 fF

Table A.2: All device sizes for the current amplifier

Device	**W** (μm)	**L** (μm)	**Multiplier**	**Area** (μm^2)	**Percentage** (%)
M1	8	1	1	10.2	2.15
M2	8	1	1	10.2	2.15
M3	12	1	1	15.4	3.22
M4	12	1	1	15.4	3.22
M5	15	2	1	34.2	7.18
M6	15	2	1	34.2	7.18
M7	10	0.750	1	1.03	2.16
M8	10	2u	1	2.28	4.792
M9	10	0.300	1	5.8	1.22
M10	10	0.300	1	5.8	1.22
M11	10	0.200	1	4.8	1.01
M12	10	0.200	1	4.8	1.01
M13	5	2.5	1	13.9	2.92
M14	5	2.5	1	13.9	2.92
M15	10	2	1	22.8	4.79
M16	10	2	1	22.8	4.79
Rz	0.500	8.13	1	4.06	0.853
Cc	5	5	9	225	47.2

Table A.3: All device sizes for the comparator

Device	**W** (nm)	**L** (nm)	**Multiplier**	**Area** (μm^2)	**Percentage** (%)
M1	10,000	150	1	2.3	32
M2	10,000	150	1	2.3	32
M3	360	180	1	0.166	1.23
M4	360	180	1	0.166	1.23
M5	2,000	150	1	0.86	6.4
M6	360	180	2	0.331	2.47
M7	360	180	2	0.331	2.47
M8	360	180	2	0.331	2.47
M9	360	180	2	0.331	2.47
M10	360	180	2	0.331	2.47
M11	720	180	2	0.662	4.93
M12	720	180	2	0.662	4.93
M13	360	180	2	0.331	4.93
M14	360	180	1	0.166	2.47
M15	360	180	1	0.166	2.47

Table A.4: All device sizes for the Banba bandgap reference. The CM devices are current mirror transistors used to generate the -200 mV reference.

	Device	W (μm)	L (μm)	M	Area (μm^2)	Percent (%)
	M1	10	2	1	22.8	0.335
	M2	10	2	1	22.8	0.335
700 mV	M3	10	2	1	22.8	0.335
	R4	0.750	16.4	32	395	5.8
200 mV	M3	10	2	1	22.8	0.335
	R4	1	58.1	4	232	3.41
-200 mV	M3	10	2	1	22.8	0.335
	CM1	10	5	1	52.8	0.776
	CM2	10	5	1	52.8	0.776
	R4	1	24	8	192	2.82
	M4	20	2	1	45.6	0.670
	M5	20	2	1	45.6	0.670
	M6	10	5	1	52.8	0.776
	M7	10	5	1	52.8	0.776
	M8	10	5	1	52.8	0.776
	M9	10	2.5	1	27.8	0.409
	M10	10	5	1	52.8	0.776
	M11	0.720	0.360	1	0.461	0.0068
	M12	0.720	0.360	1	0.461	0.0068
	M13	0.720	0.360	1	0.461	0.0068
	M14	0.720	0.360	1	0.461	0.0068
	M15	0.720	0.360	1	0.461	0.0068
	Q	17	17	1	289	4.25
	Qn	17	17	8	2,310	34.0
	R1	1	43	32	1,380	20.2
	R2	1	43	32	1,380	20.2
	R3	0.950	10.9	8	82.6	1.21

Table A.5: All device sizes for the current reference

Device	W (μm)	L (μm)	Multiplier	Area (μm^2)	Percentage (%)
M1	10	2	1	22.8	7.32
M2	10	2	1	22.8	7.32
M3	3.6	1.8	1	7.49	2.4
M4	3.6	1.8	1	7.49	2.4
M5	10	2	2	45.6	14.6
M6	3.6	1.8	1	7.49	2.4
M7	3.6	1.8	1	7.49	2.4
M8	5	2	1	11.4	3.66
M9	7.5	2	1	27.1	5.49
R1	1.2	33.8	4	162	52